DinoBank

Penetration Test Report

November 24, 2019

Table of Contents

Report Overview

Executive Summary

The following report was written on Saturday, November 23, 2019, on behalf of DinoBank as part of a limited-scope penetration test and risk assessment. The Report Overview section contains a high-level overview of our test findings, including key recommendations for improving the DinoBank corporate security posture, mitigating business risk, and reducing harm. In a subsequent section ("Technical Findings") each vulnerability is expanded upon in detail. Summaries of the primary impact of each vulnerability, as well as remediation steps, are included.

In a negative development from our prior assessment, DinoBank is vulnerable not only to external and internal security threats, but a significant degree of business risk. Such risks include compliance risk due to breaches of financial and privacy-related regulations; financial risk due to existential vulnerabilities in core, public-facing production infrastructure that have already been exploited in the wild; reputational risk due to the follow-on effects on public confidence and brand value in the event of a breach; and strategic risk if DinoBank does not adopt our strategy to become a leader in security and compliance, within its twin fields of banking and cryptocurrency.

As a banking and financial-technology (fintech) company at its core, the severity of these vulnerabilities and oversights to DinoBank's bottom line, its continued operations, and the security and confidence of its customers and employees cannot be emphasized enough. DinoBank occupies two industries, finance and cryptocurrency, in which cybersecurity concerns, public scrutiny and regulatory oversight are greater than ever. Accordingly, any breach leveraging these vulnerabilities would likely irreparably damage the organization's material and reputational value, in addition to its long-term survivability.

A recent Memorandum of Understanding (MOU) issued by the Gotham City Office of the Comptroller of Currency (OCC) has made the penalties for inaction greater than ever, directly threatening the continued existence of DinoBank. Likely insider threats, potential embezzlement, and conflicts of interest among corporate leadership, meanwhile, further degrade DinoBank's ability to rapidly and effectively act in resolution. As a result, our deliverables will feature a special focus on remedying these risks, which include a lack of security governance; weak passwords; core weaknesses exposing customer data; outdated software; poor management; and insufficient audit controls.

Fundamental security lapses in business-critical web applications and databases, such as not validating input or requiring authentication to dump customer data, threaten DinoBank's financial solvency, its brand image, and breach basic regulatory standards. These include, but are not limited to, the Gramm-Leach-Bliley Act (GLBA), which mandates important Financial Privacy and [Private Information] Safeguards Rules; the Federal Financial Institution Examination Council's Cybersecurity Assessment Tool and framework, which may be reviewed by regulators; and the upcoming California Consumer Privacy Act (CCPA), which imposes strict controls on consumers' powers over their personal data.

Importantly, as a financial institution processing cardholder data, DinoBank is also subject to PCI-DSS requirements, including Requirement 2, which restricts default password usage, and 7, which restricts

cardholder data access to a "need to know basis."[1] Laws regarding Know Your Customer, Anti-Money Laundering and Combating the Financing of Terrorism (KYC, AML and CFT)[2], such as the Bank Secrecy and USA PATRIOT Acts, also apply. DinoBank, for example, must not only verify the origin of "large sums of money" and "monitor suspicious activities," it must also "report cash transactions exceeding $10,000."[3] The vulnerabilities we have uncovered allow insider and external threats to manipulate far greater sums of money without accountability, significantly jeopardizing DinoBank's compliance efforts as the MOU reveals.

The aforementioned core infrastructure vulnerabilities allowing unauthorized access to Personally-Identifiable Information (PII) threaten customer and employee confidence and privacy, while prompting otherwise-avoidable risks of regulatory action. We firmly believe that, should a malicious actor or insider target these vulnerabilities as may have taken place, both DinoBank's future as a company, and the security of its stakeholders would be placed in jeopardy.

Ultimately, it is critical for us to emphasize that, though the DinoBank organization faces unavoidable impending risks, our security engineers and compliance and governance experts are ready and able to assist. Our remediation strategy is not only one of optimal security, but one of risk mitigation and harm reduction, anticipating risks, not simply responding to them, and skating to where the puck will be.

As such, we will collaborate not only to improve DinoBank security and protect customer and company information, but to deploy a groundbreaking security/regulatory strategy that makes DinoBank a leader in security innovation within our field. We are grateful to them for their ongoing cooperation and partnership with us, and we are proud to join them in securing the future of cryptocurrency-enabled banking.

MOU Legend

The MOU issued by the Gotham City OCC addressed the following points, according to our notes from the assessment's start date: 1) a lack of security governance; 2) weak passwords; 3) core weaknesses exposing customer data; 4) outdated software; 5) poor management; and 6) insufficient audit coverage. Throughout this report, in the style of the Mitre Common Weakness Enumeration (CWE) we will denote wherever our findings apply to compliance with the MOU by using the following legend:

1. **Lack of Security Governance -> SECGOV**
2. **Weak Passwords -> PSWD**
3. **Core Weaknesses Exposing Customer Data -> CDATA**
4. **Outdated Software -> SW**
5. **Poor Management -> MGMT**
6. **Insufficient Audit Coverage -> AUDIT**

[1] PCI Policy Portal, *PCI DSS Requirement 7*
[2] Coin Telegraph, *What Crypto Exchanges Do to Comply with KYC, AML and CFT...*
[3] Investopedia, *Anti Money Laundering (AML)*.

Scope

Observation of Scope

The scope of this penetration test was the CIDR ranges 10.0.1.0/24, 10.0.2.0/24, 10.0.10.0/24, 10.0.11.0/24, and 10.0.12.0/24. The scope also included a Hyosung ATM and IVR system.

Our security engineers were careful to adhere exactly to this scope. In addition, as we were testing production systems, we took care not to disrupt production services, exfiltrate private customer information, or modify or delete production data. Any modifications made to systems during the course of the engagement were removed by our security engineers at the termination of the engagement period. Furthermore, upon accessing sensitive customer information, our engineers took care to not exfiltrate data beyond DinoBank networks and to only analyze customer metadata.

Overall Observations

Insufficient Access Controls and Authentication Policies (MOU Legend: PSWD, CDATA, AUDIT)

In continuation from our last assessment, one of the primary vulnerabilities within the DinoBank infrastructure was still insufficient or missing access controls in web applications and databases. Several systems, including databases containing customer and employee PII and banking information, allowed unauthorized users to access and modify their information and configuration. As a result, our engineers were able to access extremely confidential information, from customers' bank account balances and card PINs, to their Social Security/tax ID numbers, names and phone numbers. This was primarily due to unsafe unauthenticated access and relatively simple security misconfigurations.

Not only do insufficient access controls negate DinoBank compliance with financial industry and data-protection standards such as PCI-DSS and GLBA, they place customers and employees at significant risk of identity theft, spear-phishing, and blackmail. Accordingly, DinoBank may be negligently liable if malicious actors were to obtain this information, incurring hundreds of thousands of dollars in potential fines per violation[4]. Additionally, DinoBank may lose untold public trust in two industries (banking and cryptocurrency) with histories of experiencing data breaches, even as they hold the public's most sensitive, life-essential data.

Steps to remedy this issue include disallowing unauthenticated access to DinoBank databases; enforcing strong, NIST framework-compliant password policies utilizing hashing, rotation, lockouts and salting along with encryption standards such as elliptic-curve cryptography, RSA and Diffie-Hellman; and removing hard-coded and excessively simple credentials from company systems and APIs. No application should be permitted to allow access with default credentials, no credentials, or guest credentials.

Furthermore, sensitive data should be encrypted both at-rest and in-transit using HTTPS/TLS, SSL certificates, and NIST-compliant cryptography. Audit policies should be strictly enforced to monitor attempts to access and

[4] FICLaw: Privacy Law Basics: Don't be Glib...

update critical databases, and such systems should monitor for fraudulent, unusual and exceptional transactions reportable under KYC/AML laws. Finally, logins should necessitate multi-factor authentication, preferably utilizing push notification-based or TOTP-based services and not SMS, which is vulnerable to SIM swapping attacks.

Remedying these issues would address MOU points regarding DinoBank's lack of security governance (by implementing standards in accordance with NIST and FFIEC Cybersecurity/Risk Management Framework recommendations); weak passwords and core weaknesses jeopardizing consumer data (by strengthening password policies and authentication); and insufficient audit coverage (by strictly monitoring who is able to access such confidential data).

Weak Password Policies and Critical Information Stored in Plaintext (MOU: PSWD, CDATA)

Furthermore, passwords were stored in plaintext in both Windows servers and the critical PostgreSQL database. Due to lax password and authentication policies, users implemented weak passwords which allowed our engineers easy access into the Windows systems. Such weak user passwords, in conjunction with a lack of encryption of both PII and other administrative credentials, facilitated our engineers' gaining greater privileged access to DinoBank systems.

Such insecure plaintext credential storage and password policies could directly cause financial loss for customers and possible identity theft. This may cause massive loss of trust in DinoBank, as well as a plethora of liability and compliance issues that would place DinoBank in a highly tenuous position given the current MOU. To ameliorate this problem, DinoBank should hash and salt all stored passwords using industry-standard slow hashing algorithms, enforce strong password rules for all employees and customers, and encourage or preferably require multi-factor authentication.

Insufficient Input Validation (MOU: CDATA)

Our security engineers, working on behalf of DinoBank, were also able to leverage insufficient validation and sanitization of user input to modify legitimate HTTP requests without authorization. This would allow malicious actors, in one case, to effectively generate limitless funds and conduct unauthorized transactions without customers' consent, all by simply exploiting publicly-available resources. To mitigate such attacks, both public-facing and internal DinoBank infrastructure (especially the core, sensitive banking application) must sanitize and validate input and transaction attempts. This would ensure, as we accomplished, that customers are unable to send negative amounts of money or transfers to themselves, and cannot transfer funds from accounts that are not theirs. As such, both customers and the company would be protected, ensuring they remain financially solvent and in control of their own funds.

Insider Threats and Inadequate Management Policies (MOU: SECGOV, CDATA, MGMT)

During our assessment, we experienced a number of encounters, where individuals at various levels of the DinoBank organization requested us to bypass corporate policies, violate our agreements, disclose our findings privately to them, work against internal employees, and help disguise potential fraud or embezzlement by prioritizing or de-prioritizing certain systems or members of leadership. These included multiple direct requests to ignore or contravene our legitimate chain-of-authority to benefit a potential fraudster or spread

potential malware or abuse to corporate systems. Finally, some requests employees made of us seemed to indicate that different employees were not operating on a common understanding of our activities.

We reiterate that at no point did we ever oblige illegitimate requests, and we worked to clear confusion whenever possible. However, the fact that several employees see such conduct as acceptable, or hold inconsistent views of security activities, belies a great need to enforce strong management policies and clear security governance practices. Such policies may include, for example, an anonymous insider threat and waste, fraud and abuse reporting hotline, and strong penalties, up to and including prosecution or termination of employment, for unlawful, unacceptable or fraudulent conduct.

One particular case of likely insider threat and fraud was observed in the testing ATM's transaction histories. Our security engineers observed a transaction for an account with balance over $99 million. Upon further investigation, this yielded an account with balance close to $1 billion, belonging to a user registered with email domain "teamblack.co". Even more concerning was the fact that the account was created just two days ago, and that the account's balance was highly suggestive of having been manipulated. Based on analysis of this user's activity and password, it appears that the user demonstrates an affinity for croissants, which is not inconsistent with behavior of DinoBank executive Alex Faulkner.

Inadvisable Security Practices (MOU: SECGOV, MGMT)

During initial reconnaissance before the start of the engagement, our security team encountered several concerning security practices conducted by employees of DinoBank. First, social media posts on Twitter and Instagram by DinoBank employees led our researchers to discover a public eBay transaction in which employees purchased twelve Hyosung 1500 ATMs on September 25th, 2019, and subsequently transported these ATMs by Penske truck from Mason, Ohio to the site of engagement. This degree of information leakage is dangerous to DinoBank, as it provides competitors insight into new DinoBank infrastructure and services before DinoBank has made this information public.

Additionally, our researchers identified a private SSH key posted to the public GitHub account of a DinoBank employee, which presents a large security concern, as it would allow malicious actors to login to DinoBank networks as this user. We notified this employee by email over a month before this engagement, but the private SSH key remains publicly exposed as of the writing of this report.

Finally, there were several concerning internal security practices that our researchers encountered during the course of the engagement. Most notably, a DinoBank employee managing an ATM left the management console open upon completing their diagnostic tests. It is necessary that critical infrastructure such as ATMs be appropriately secured whenever not actively in use by a DinoBank employee with management permissions.

High-Level Recommendations (MOU: All Points)

In sum, we recommend several major remediations that would significantly boost DinoBank's security posture, mitigate regulatory risk, better protect customers' and employees' critical data and safeguard the future of the company. Our first suggestion is to, as aforementioned, implement strong, zero-trust access controls and authentication policies, from rigid password policies to multi-factor authentication to encryption at rest and in transit. Another is to implement basic input sanitization and validation. As part of a layered, defense-in-depth strategy, these measures would significantly impede adversaries' attempts to exploit common attack vectors,

maximizing the cost they would incur to attack bank infrastructure while minimizing DinoBank's own cost of maintenance and legal risk.

Furthermore, DinoBank should work to clarify, standardize and strengthen its security governance and management policies, as emphasized by the MOU. A critical part of any security governance strategy is a risk matrix and a set of common standards; the FFIEC has provided such in their IT Information Security Handbooks and Cybersecurity Assessment Tool, and so has NIST in their Cybersecurity Frameworks.

Yet standards and guidelines cannot work without effective management to implement them. As such, these policies must be implemented in a visible, tangible manner. In addition to the aforementioned examples, improvements may include background and reference checks on all employees, and psychological evaluations on employees handling high-risk infrastructure and data, similar to Department of Defense standards. As first steps in a comprehensive security and governance strategy, their remediations will significantly secure both DinoBank infrastructure and its employees' and customers' PII.

Moreover, we recommend that DinoBank follow industry-standard procedures and establish a vulnerability disclosure policy in order to allow receiving vulnerability reports from external security researchers. Financial organizations such as JP Morgan Chase and Capital One, among others, have established such policies and understand that transparency is key to ensuring ongoing security success. Moreover, the FTC has alleged that failing to list a public contact channel for vulnerability reports constitutes an "unreasonable practice, in violation of Section 5 of the FTC Act" (8). We recommend that DinoBank follows suit in order to enable channels for hearing from researchers with potentially critical information.

As an illustration of the need for a vulnerability disclosure policy, one of our engineers discovered a publicly exposed RSA key of DinoBank on an employee's GitHub account prior to the start of the engagement. While we attempted to contact the employee directly, a vulnerability disclosure policy would have enabled swift action and revocation of that secret key.

DinoBank at present is utilizing ATM systems that are outdated and incompatible with modern security, data transport and encryption standards such as TLS 1.3 and the EMV chip-based payment method. As such, we recommend upgrading the ATMs to comply with modern regulatory and technological requirements as soon as possible rather than using old or retrofitted solutions. Such systems will almost certainly be included in consideration as part of evaluations of DinoBank's standing under the MOU.

Lastly, we note that we discovered no indication of a backup or version-control system deployed on DinoBank. Such controls are necessary for smooth operation of any technology organization and may aid in the event of data loss or targeted attacks. We recommend that DinoBank implement multiply-redundant on-site and off-site backups in order to mitigate the risk of data loss.

Positive Security Controls (MOU: PSWD, CDATA, SW)

The DinoBank network contained a number of effective security mechanisms. For starters, the vast majority of public-exposed operating systems, services, and programs were up-to-date or nearly up-to-date, which rendered many popularized and simple exploits (e.g. ETERNALBLUE for Windows) infeasible. Additionally, a number of basic security best practices were adopted in specific locations, limiting unauthorized access. The following is a series of examples demonstrating the aforementioned best practices:

- SMB message signing was always required.
- The SMTP client was secured through NTLM authentication, preventing spoofing and impersonating another user.
- All SSH endpoints utilized key-based access, so that even an attacker with credentials is impeded from easy access.
- SQL queries were generally parameterized, reducing the possibility of SQL injection attacks.
- The impact of anonymous FTP access was limited by read-only permissions.
- The physical ATM did not use default management codes, limiting operational access.
- Brute force enumeration of domain credentials was severely restricted by the application of a strict account lockout policy.
- The Asterisk API used for the Interactive Voice Recognition system did not use default credentials and produced no positive hits during brute force attempts on credentials.

In addition to the above already well established security practices, some vulnerabilities which were present in the previous engagement were fixed before this one. These are:

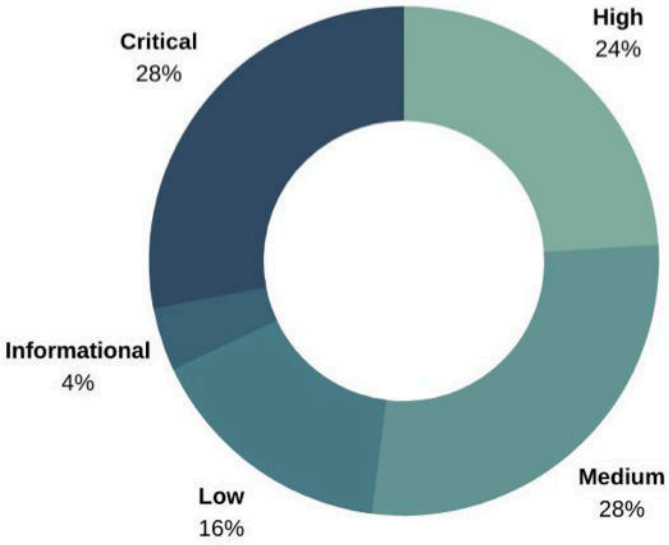
- The public-facing restaurant site that was previously editable by all users was removed, thus eschewing all worry about the site's defacement.
- The PostgreSQL user 'bank' on machine 10.0.2.100, which previously granted unauthenticated access to the database, was removed.

The above positive security controls significantly reduced the efficacy of attacks against DinoBank, while providing a foundation for future improvements.

Technical Findings

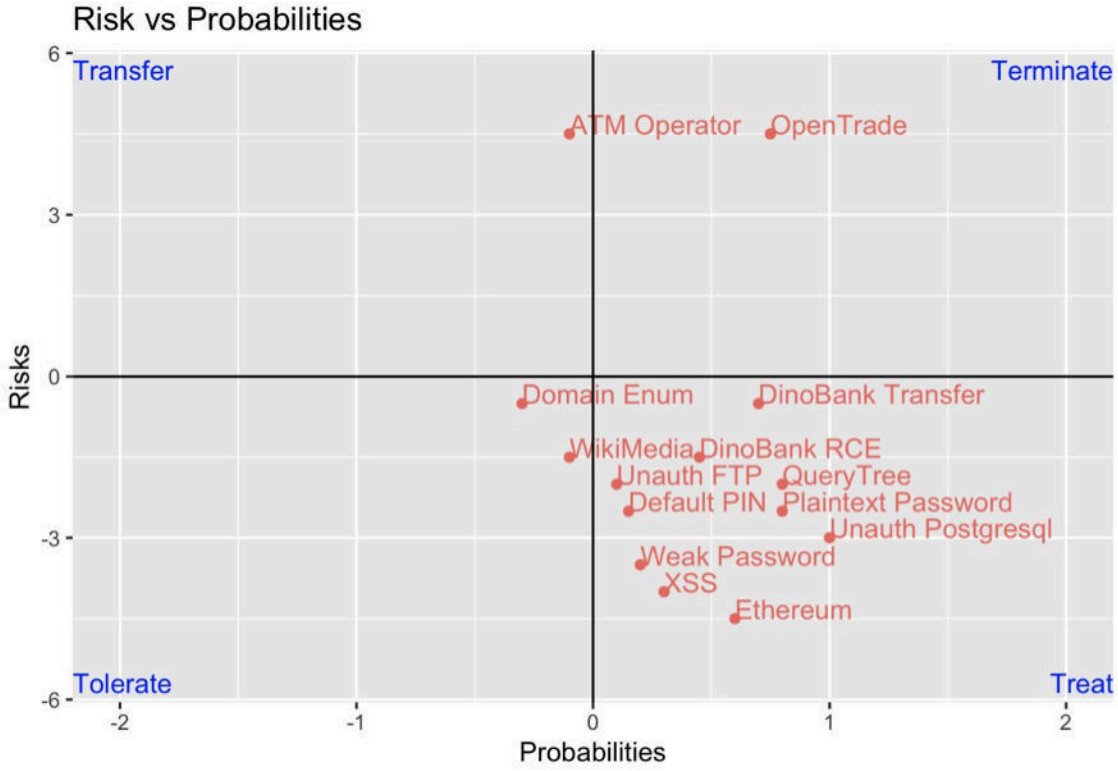
The following pie chart summarizes the severity of vulnerabilities found during the penetration test. These vulnerabilities are categorized by risk level, as calculated under the Common Vulnerability Scoring System (CVSS).

Breakdown of Risk Levels for Found Vulnerabilities:



The following chart details discovered vulnerabilities based on risks and associated exploitation probabilities.

Terminate, Transfer, Tolerate, Treat Graph



Critical Risk

Unauthenticated PostgreSQL Database Exposes Customer, Company Information (MOU: PSWD, CDATA, AUDIT)

Threat Level: **Critical (9.6)**

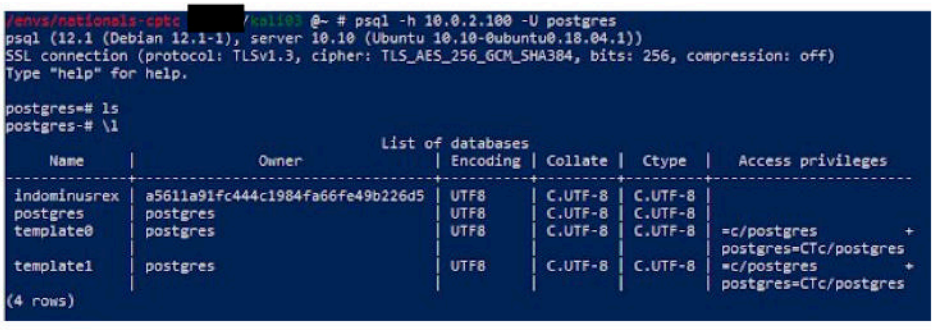
Description:
DinoBank's core PostgreSQL database is accessible without authentication, allowing network visitors to access and modify complete database records.

We note that this exploits a vulnerability previously reported in the initial penetration report, although using different database users.

Potential Business Impact:
Unauthenticated access to the PostgreSQL database exposes sensitive corporate and customer information, as prohibited by the MOU, including bulk PII exposure of over 20,000 DinoBank customers, such as names, taxpayer IDs (Social Security numbers or employer identification numbers), phone numbers, email addresses, home addresses, and bank pins. Other accessible data includes employee accounts and plaintext passwords, customer accounts and plaintext passwords, transaction history, and other account-related information. Any such data can also be modified or deleted at will, leading to disruption of DinoBank's services. Note that the taxpayer IDs and pins are also the same credentials used to access the IVR systems, which are a core part of DinoBank's services. Such access must be subject to stringent audit and access control policies to prevent data breaches.

Affected Host:
10.0.2.100 -- CORE-01

Exploitation Details:
By connecting to the PostgreSQL database at port 5432 on the host 10.0.2.100 with username 'postgres', our security engineers were able to gain full access to the database.



Gaining access to the PostgresQL database via the "postgres" username

As can be seen in the screenshot above, a security engineer gained unauthenticated access to the bank database via the user ‘postgres’ without being prompted for a password. The postgres user is the highest privileged user of the database.

From within the database, our security engineers were able to identify the personal information of more than 360 DinoBank employees and 8,000 DinoBank customers, including taxpayer IDs (Social Security numbers or employer identification numbers), phone numbers, email addresses, home addresses, and bank PINs. In addition, our security engineers were able to exfiltrate each of those customers passwords, as they were stored in plaintext within the database. Both of these are shown below (with redactions).

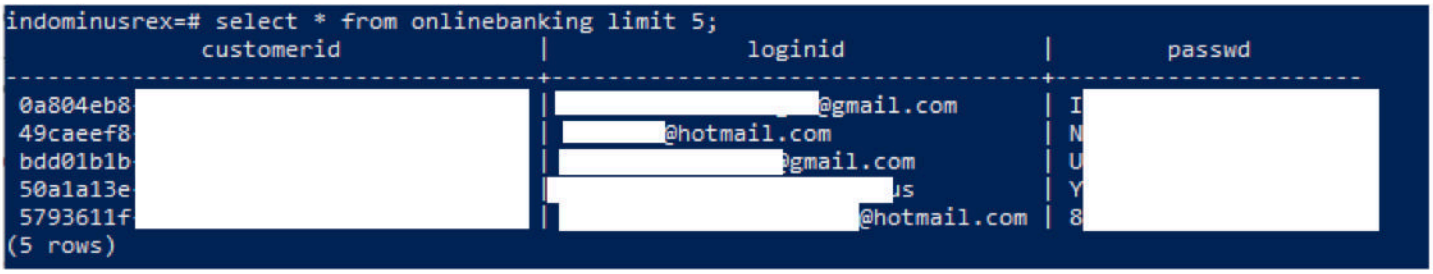


Table containing employee PII, customer logins, and plaintext passwords

Finally, our security engineers were able to use the taxpayer IDs and bank PINs shown to access the IVR system and retrieve user account information.

Remediation Recommendations:
The PostgreSQL database should require authentication by requiring a password that follows strong complexity standards. Since there also exists an unauthenticated user ‘postgres’ which can access all but one table, requiring a complex password to authenticate the user ‘postgres’ would also greatly mitigate risks. A thorough investigation to see whether any other services in DinoBank also lack password authentication would mitigate these simple attacks.

References:
PostgreSQL Security Documentation

Remote Command Execution on PostgreSQL Database (MOU: CDATA)

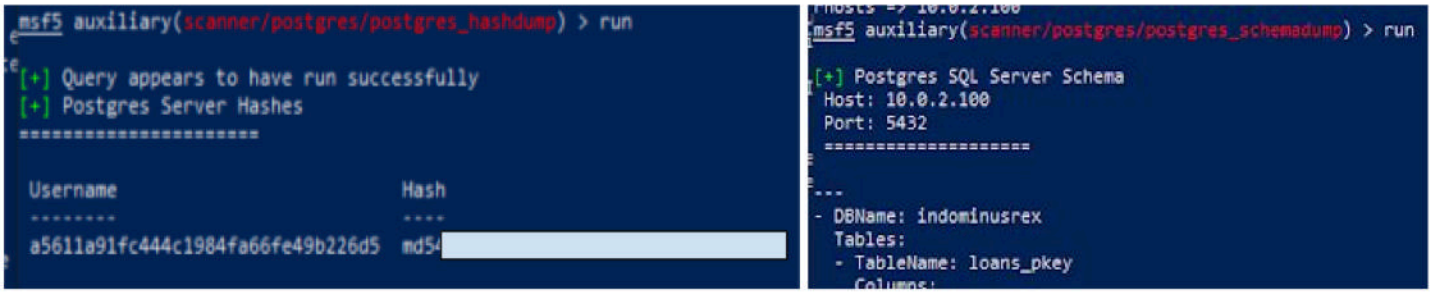
Threat Level: **Critical (9.6)**

Description:
After gaining access to the unauthenticated PostgreSQL database, it was observed that SQL access could be taken advantage of via several Metasploit exploits, allowing privilege escalation and full command execution. This allows the execution of arbitrary code on the PostgreSQL server and further potential pivoting to connected services. Note that this vulnerability was originally included in the initial penetration test report and remains unremediated.

Potential Business Impact:
This allows an unauthenticated attacker to execute arbitrary commands on the PostgreSQL host. After attaining command execution, we observed sensitive information hosted on the server, such as API service source code and DevOps configurations, which would allow for the exposure of the intellectual property (IP) of DinoBank. Furthermore, an attacker could potentially gain privileged access on the host, allowing the manipulation of data or further escalation of privileges.

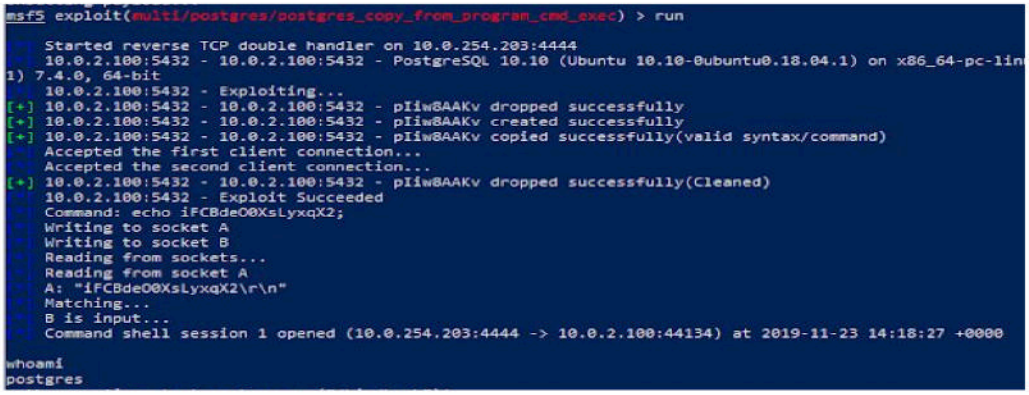
Affected Host:
10.0.2.100 -- CORE-01

Exploitation Details:
Our security engineers began by scanning the PostgreSQL database for further information. Two Metasploit scans yielded information on username/hash pairs and the schema of the database tables, as shown below.



Two Metasploit scans returning sensitive database information

Our security engineers then succeeded in gaining shell access to the target in question. Since 'postgres' has administrative privileges within the postgreSQL environment, a Metasploit exploit was run to obtain a shell on the host box, and various files were obtained and analyzed. These files included source code for DinoBank's core API service and records of API endpoints.



Attaining a shell as user 'postgres' on the target machine via Metasploit

Furthermore, a config file of the NodeJS application contained a redis username and password for the croissantco.in domain on port 6379 (pictured below). We did not validate these credentials further as upon following up with DinoBank leadership, the domain was confirmed out of scope for the current engagement.

Configuration file containing Redis password

Remediation Recommendations:
Adding authentication to PostgreSQL login as described in the previous finding will help mitigate this issue. Furthermore, to provide more security, the company should adopt a defense-in-depth approach to protect their critical data. By keeping databases on isolated systems and networks, many intrusion vectors are severely restricted. Finally, a thorough analysis of the database is necessary to ensure that there is no further data that can be exfiltrated or vectors for privilege escalation.

References:
Granting Role Membership
Converting between User and Superuser
Network Segmentation Best Practices

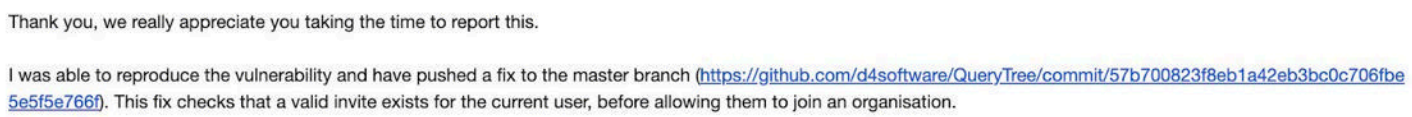
Access Control Bypass Allows Administrative Access to QueryTree (MOU: CDATA, SW)

Threat Level: **Critical (9.4)**

Description:
There exists an underlying vulnerability in QueryTree allowing unauthenticated visitors to join arbitrary QueryTree organizations as administrators. This entirely bypasses the QueryTree invitation process, and allows access to DinoBank's configured QueryTree instances.

Note that this represents an underlying vulnerability in the open source QueryTree software, and thus affects any deployed QueryTree instance. As per our disclosure policy, we have contacted the vendor with technical information to allow remediating the vulnerability.

Our engineers disclosed this vulnerability to D4 software, the vendor maintaining QueryTree, shortly after its discovery. The vendor issued a patch for the vulnerability the day of reporting and the vulnerability is pending CVE.


Thank you, we really appreciate you taking the time to report this.

I was able to reproduce the vulnerability and have pushed a fix to the master branch (https://github.com/d4software/QueryTree/commit/57b700823f8eb1a42eb3bc0c706fbe5e5f5e766f). This fix checks that a valid invite exists for the current user, before allowing them to join an organisation.

Response from D4 software confirming receipt of vulnerability report

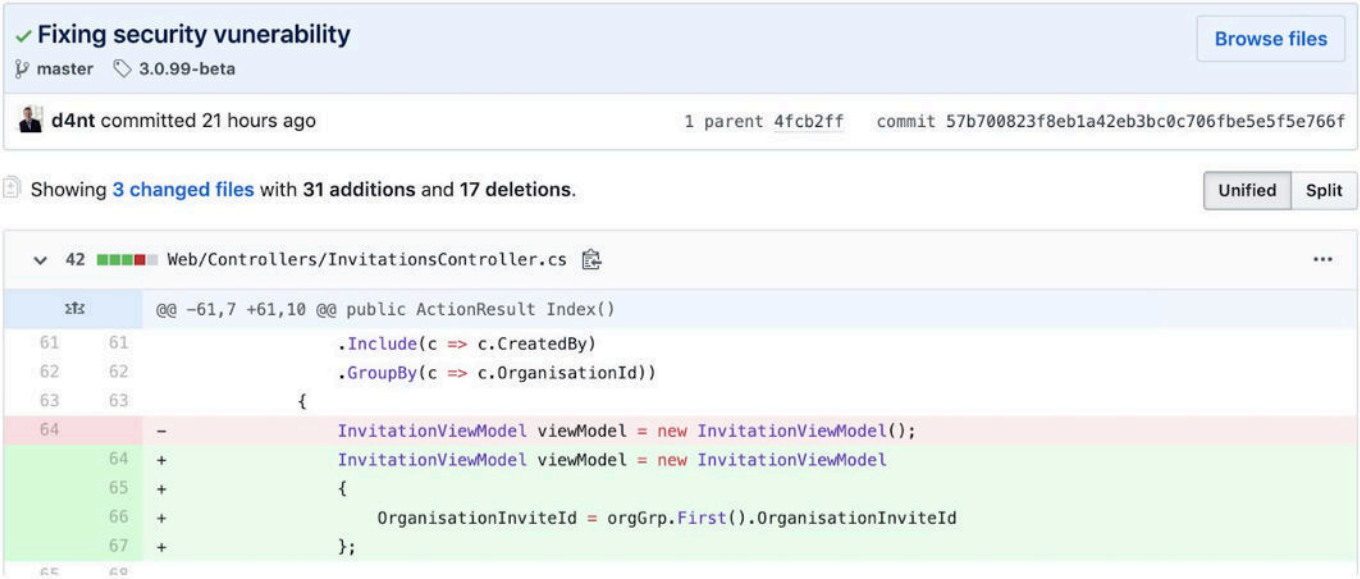


Image of patch issued by QueryTree as a result of our disclosure

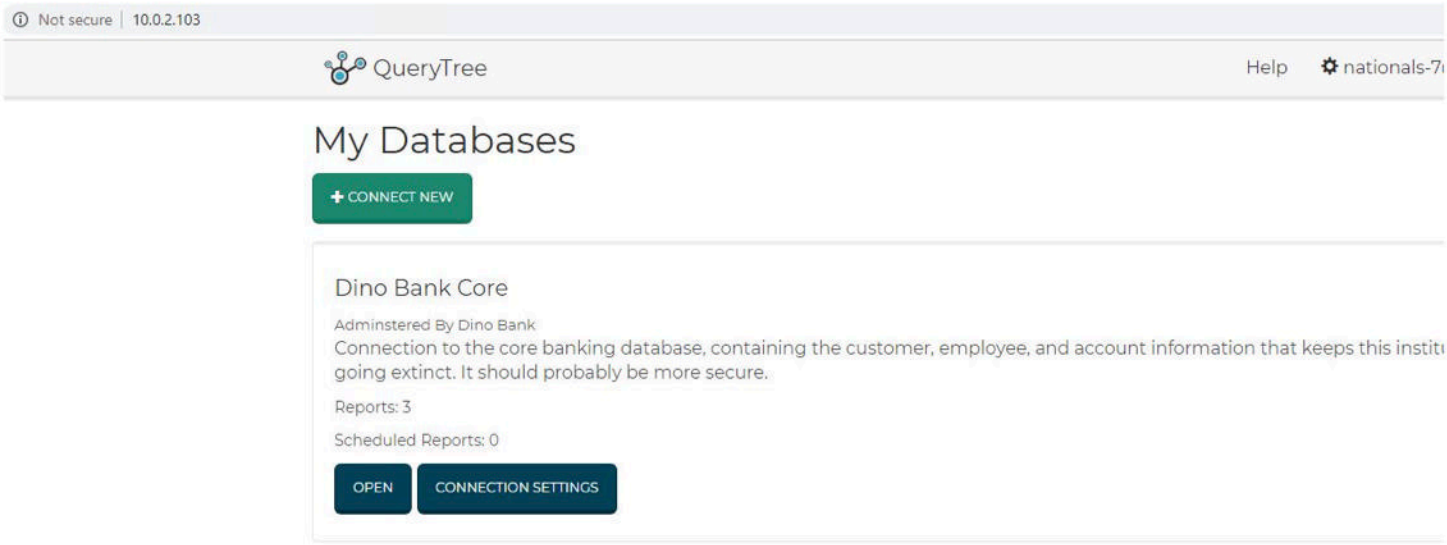
Potential Business Impact:

This allows an unauthenticated attacker to gain complete access to arbitrary registered QueryTree organizations simply by registering an account. In the case of DinoBank, this exposes one registered organization, which is linked to a PostgreSQL database.

Affected Host:
10.0.2.103 - REPORTS-01

Exploitation Details:
In order to reproduce, visit QueryTree at 10.0.2.103 and register for an account.

Then, visit http://10.0.2.103/Invitations/Accept/1. This exploits a vulnerability in QueryTree to force joining the core DinoBank account. Observe that the attacker's account will be added as an admin to that organization without validation.



Administrative access to core DinoBank QueryTree database

The patch issued by QueryTree ensures that the Accept function of InvitationsController.cs validates that a user has an invitation to the given organization before adding the user to the organization.

Remediation Recommendations:
In order to resolve this vulnerability, DinoBank should update its QueryTree instance with the latest patch as soon as possible.

Unauthenticated SQL Injection in OpenTrade Via API (MOU: PSWD, CDATA, SW)

Threat Level: **Critical (9.4)**

Description:
There exists an underlying SQL injection vulnerability in OpenTrade allowing execution of arbitrary SQL queries. This can be exploited to access arbitrary information in the OpenTrade database, such as account details, trade histories, and session tokens.

Note that this represents an underlying vulnerability in the open source OpenTrade software, and thus affects any deployed OpenTrade instance. As per our disclosure policy, we have contacted the developer with technical information to allow remediating the vulnerability.

Our engineers disclosed this vulnerability to the developer maintaining OpenTrade shortly after its discovery. The developer issued a patch for the vulnerability the day of reporting and the vulnerability is pending CVE.

Image of patch issued by OpenTrade as a result of our disclosure

Potential Business Impact:
This allows an unauthenticated attacker to gain complete access to the OpenTrade database, including account information, trade histories, and session tokens. Furthermore, given access to administrator session tokens, an attacker can login with the administrator account, allowing complete control over the OpenTrade instance.

Furthermore, note that there are two reports, linked here and here, which cumulatively report the theft of over $1 million via SQL injection in live cryptocurrency exchanges deploying OpenTrade. It appears likely that the vulnerability that we have found here is the vulnerability that is being exploited in those reports. Since this would mean that this vulnerability is actively being exploited in the wild, we recommend that DinoBank immediately suspend operation of OpenTrade until such vulnerabilities can be resolved and a thorough security review conducted.

Affected Host:
10.0.1.250 - COINS-01

Exploitation Details:
In order to confirm this vulnerability, first visit http://10.0.1.250/api/v1/public/getmarketsummary?market=MC-LTC". Observe that an "unknown database error" is returned, demonstrating the SQL error.

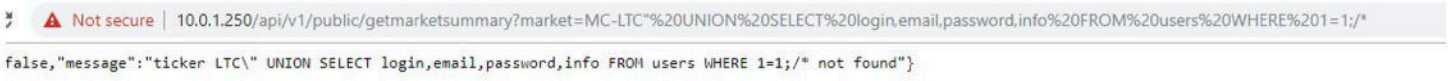
SQL database error indicating escaping SQL query

Next, visit http://10.0.1.250/api/v1/public/getmarketsummary?market=MC-LTC"" with two quotes. Observe that an error is not returned as the two quotes escape each other.

Information can be exfiltrated via blind methods. For instance, the following query tests if a user exists whose email begins with "test":
http://10.0.1.250/api/v1/public/getmarketsummary?market=MC-LT%22%20UNION%20SELECT%20login,email,password,info%20FROM%20users%20WHERE%20%22login%22%20LIKE%20%27test%25%27;/*

This works by joining the query with another query to select user information conditioned on a wildcard operator.

Using this, we confirmed that there existed no users in the OpenTrade database. However, if such users existed, this would allow exfiltrating their account information, such as password hashes, trade history, and session details.



Response indicating that zero users exist in the OpenTrade database

Remediation Recommendations:
In order to resolve this vulnerability, DinoBank must update its OpenTrade instance with the latest patch as soon as possible. Due to the high risk associated with SQL injections in this software, our security engineers recommend implementing the Dinobank cryptocurrency exchange with a different service.

Remote Command Execution via Profile Image Upload (MOU: CDATA)

Threat Level: **Critical (9.4)**

Description:
The profile image upload functionality of the core DinoBank web application fails to properly validate image file types before uploading. As a result, an attacker can execute arbitrary remote commands by uploading a malicious PHP file.

Potential Business Impact:
Any DinoBank user may execute arbitrary commands on the core DinoBank web server. This allows, for instance, arbitrary access to the DinoBank core database (via an intermediate API), and modification of existing application functionality. This could be exploited to access DinoBank customer information in bulk or serve malicious content from DinoBank.

Affected Host:
10.0.2.101 -- BANKWEB-01 (my.dinobank.us)

Exploitation Details:
After creating an account at my.dinobank.us, our engineers proceeded to edit the account at account.php. While uploading an image, our engineers modified the image extension to ".php" and inserted PHP code into the contents of the image:

Uploading a PHP reverse shell

Our engineers then visited https://my.dinobank.us/img/user. This returns a list of uploaded images to the server. Our engineers visited the uploaded PHP file and observed that the PHP script executes.

```
?>www-data@bankweb-01:/var/www/html$ ls
ls
LICENSE
account.php
accounts.php
alert.php
close.php
commiter.sh
config.php
crypt.php
css
dashboard.php
data.csv
db.php
functions.php
git.zip
img
index.html.bak
index.php
js
loans.php
login.php
logout.php
navbar.php
new.php
notary.php
phpinfo.php
process.php
register.php
scss
sessman.php
sidebar.php
transfer.php
vendor
www-data@bankweb-01:/var/www/html$ cat config.php
cat config.php
<?php
// disable errors
error_reporting(0);

define('APIPORT',443);
define('APIURL','https://bankasaurus.dinobank.us/v1');
define('APIKEY','58515E26[redacted]');
define('BANKNUMBER','');
?>
```

Executing commands via a PHP reverse shell

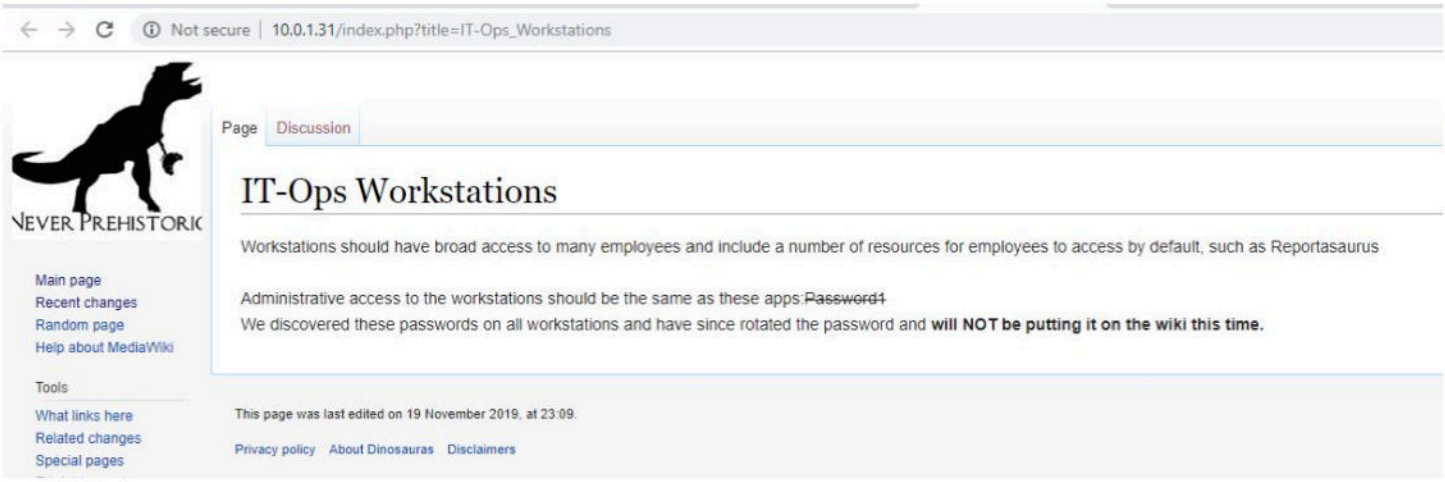
From here, arbitrary commands can be executed on the core DinoBank server. In particular, this led to the discovery of an API key, pictured above, used to connect to the DinoBank API and make arbitrary database requests.

Remediation Recommendations:
Validate the content type and extension of uploaded files to prevent uploaded files from being treated as code.

Weak Password for Local Admin Credentials (MOU: PSWD)

Threat Level: Critical (9.4)

Description:
An outdated page of the company wiki listed previous default passwords for local administrator accounts on Windows workstations on all three branch networks (“Password1”, see screenshot). Though the passwords had since been changed, a similar password (“Password2”) allowed local administrator access to ten Dinobank workstations that possessed sensitive credentials.



Wiki post on default local administrator credentials

Potential Business Impact:
These workstations are joined to the same domain as the critical banking infrastructure, allowing these hosts to enumerate critical information about the domain (including the domain administrator accounts, see Domain Admin Enumeration). If these workstations were used as a foothold for full domain compromise, it would allow the attacker to make arbitrary changes to both internal and public-facing corporate infrastructure. Any resulting service downtime could lead to massive financial loss for the company. Furthermore, the exfiltration of employee and customer data could lead to critical regulatory violations.

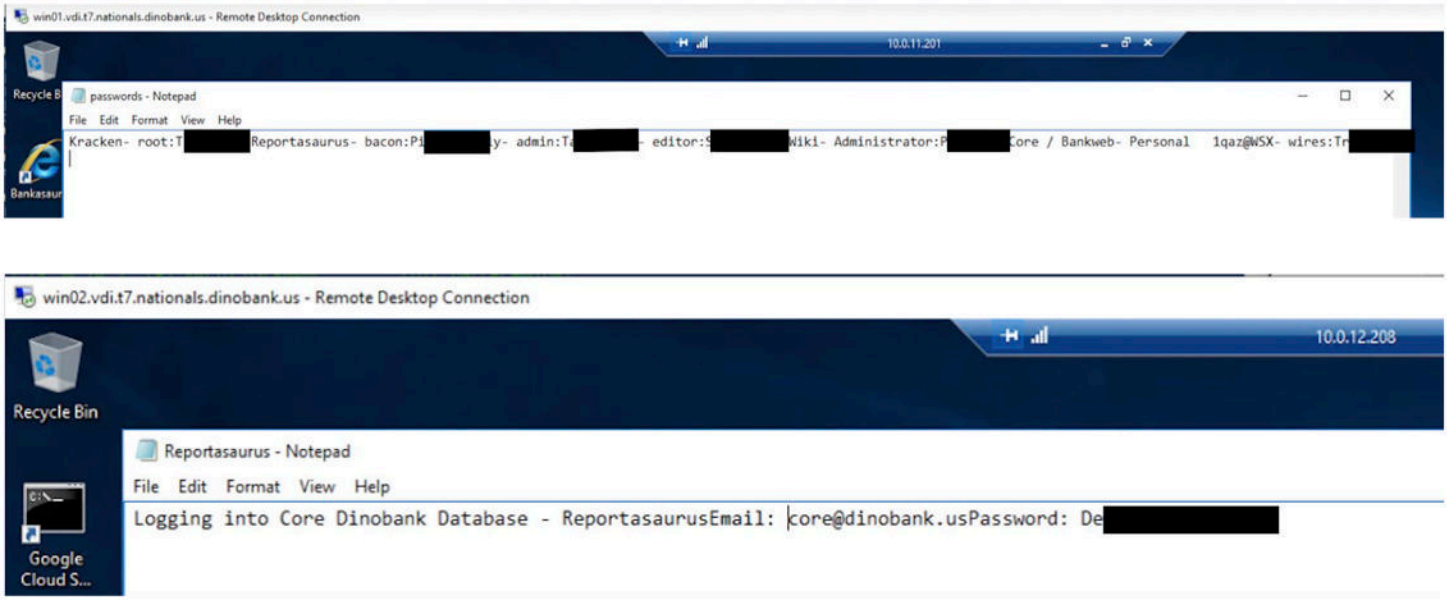
Affected Hosts:
Gotham Branch
10.0.10.201
10.0.10.202
10.0.10.203
10.0.10.208
10.0.10.209

Metropolis Branch
10.0.11.201
10.0.11.202

10.0.11.208

Springfield Branch
10.0.12.201
10.0.12.208

Exploitation Details:
The use of the password “Password2” with the user “Administrator” allowed for authentication and access to a total of ten hosts in the three branch networks. A session could be established using the Remote Desktop service enabled on each of these hosts. Several of the hosts had sensitive credentials stored in plaintext on their desktops (see below figures). These credentials included access to the core Dinobank database utility on 10.0.2.103, the administrative account on the wiki on 10.0.1.31, and user credentials on several other hosts in the internal bank network (10.0.2.101, 10.0.2.103, 10.0.2.113, 10.0.2.115).



Plaintext credentials stored on Administrator desktop (10.0.11.201, 10.0.12.208)

Remediation Recommendations:
Set unique, high-entropy passwords on all workstations using a secure password generator. Do not store any credentials in plaintext on hosts, even if they are intended only for internal use.

References:
Secure password generation
Password best practices, 2019

Publicly Exposed Ethereum JSON-RPC API Server with Default Password (MOU: PSWD)

Threat Level: **Critical (9.1)**

Description:
As detailed in the initial penetration test report, an administrative service was discovered on DinoBank's CroissantCoin website exposing an Ethereum JSON-RPC API server with a weak default password. Such services are used by Ethereum clients to interface with wallets, and allow taking actions using that wallet's private keys, such as viewing address information, signing messages, and sending transactions. Public exposure of such interfaces allows any visitor to interface directly with DinoBank's core cryptocurrency wallets.

Due to its severity, we disclosed this vulnerability to Tom Dickson ahead of the engagement. We note that this vulnerability now appears to be remediated.

Potential Business Impact:
Several accessible Ethereum addresses indicated that private keys were stored on the server in question. Furthermore, given the usage of a weak default password, it is possible to compromise all wallets, leading to irreversible financial loss for DinoBank. The wallets contained over 1,375 ETH collectively, valued over $240,000 at the time of writing.

Affected Host:
http://croissantco.in - Note that this host was out of scope during this current engagement. We include it in this report only for full disclosure, as our engineers discovered this vulnerability during reconnaissance prior to the engagement and the vulnerability is a severe one.

Exploitation Details:
It was observed that visiting https://croissantco.in/admin/index.js exposed an admin JavaScript file that connected to port 8543 on the local host over Web3. Port 8543 was exposed publicly, allowing any internet visitor to establish connections to the RPC service. We verified this by locally running Web3 and connecting to the service, where we confirmed that address information could be obtained and key passphrases tested. Furthermore, access could be obtained to send and sign transactions given the weak default password of "123456".

We note that this vulnerability was discovered during the initial penetration test engagement, and it was shortly after the report was delivered that we noticed the server was configured with a weak default password of "123456". Due to the severity of this vulnerability, we immediately contacted Tom Dickson with steps to reproduce the vulnerability.

```
const Web3 = require('web3')
const rpcUrl = 'http://croissantco.in:8543'
const web3 = new Web3(rpcUrl)

const coinOwner = '0x3cC1c22c86eD77b2E63BF6aCDAf25d78E8f9Ad95' // CROX contract owner

web3.eth.getBalance(coinOwner, (err, wei) => {
  balance = web3.utils.fromWei(wei, "ether")
  console.log(balance)
})
```

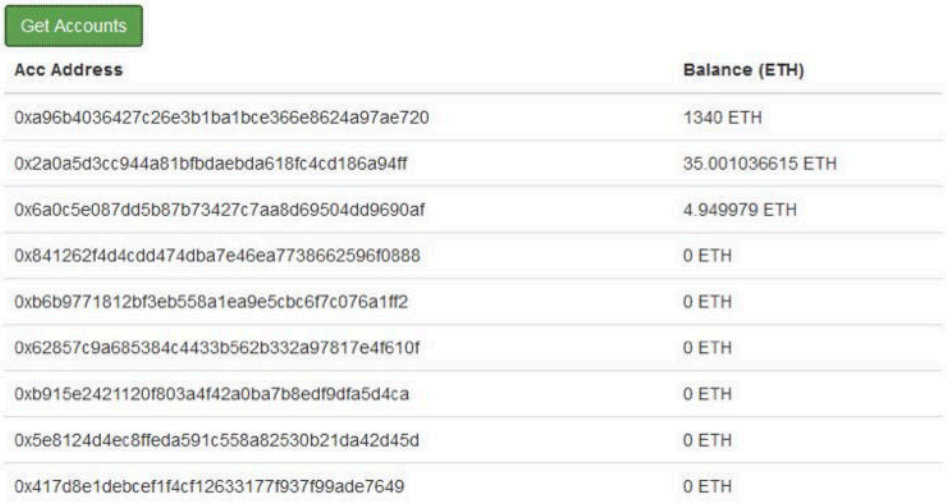
Script written to access address information by connecting to the RPC service

```
# SIGN REQUEST
POST / HTTP/1.1
Host: croissantco.in:8543
Content-Length: 233
Pragma: no-cache
Cache-Control: no-cache
Origin: http://croissantco.in:8000
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_3) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/77.0.3865.90 Safari/537.36
DNT: 1
Content-Type: application/json
Accept: /
Referer: http://croissantco.in:8000/
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9,la;q=0.8
Connection: close
{"jsonrpc":"2.0","id":5,"method":"eth_sign","params":["0xa96b4036427c26e3b1ba1bce366e8624a97ae720","0x
b7547e727eaca365fdc44bd057dd4b5838c65cb6818a6071e6b3d4e6c413bed906fe933e5f3aca4e65650242e59223b8b22ff5
0bc3e2fd12a8c171b8b4e8a525"]}

# SIGN RESPONSE
HTTP/1.1 200 OK
Access-Control-Allow-Origin: http://croissantco.in:8000
Content-Type: application/json
Vary: Origin
Date: Sat, 19 Oct 2019 23:18:02 GMT
Content-Length: 169
Connection: close
{"jsonrpc":"2.0","id":5,"result":"0xac6d8a5f4948c46aee103099705e81f080bef1205e4b0fdb2d6bcd1fec4270cd69
004accb674f464600fa3f824e91490a4b8a451e4e647db5d847c9a802721201c"}
```

Request made to the RPC service to sign a randomly-generated message using the server's private key

By accessing the RPC service, at least 9 addresses appeared to be managed by that server's wallet. This is in line with the Ether explorer accessible at http://croissantco.in:8000, which listed addresses and associated balances.



Listed balances for accessible addresses

Remediation Recommendations:
It appears as if the exposed server is no longer accessible. DinoBank should ensure that at no point do such servers become publicly accessible again, as they risk exposing cryptocurrency funds to the public.

References:
[Protecting Ethereum JSON-RPC API with password]

High Risk

Improper Validation of Account Transfer Amounts (MOU: AUDIT)

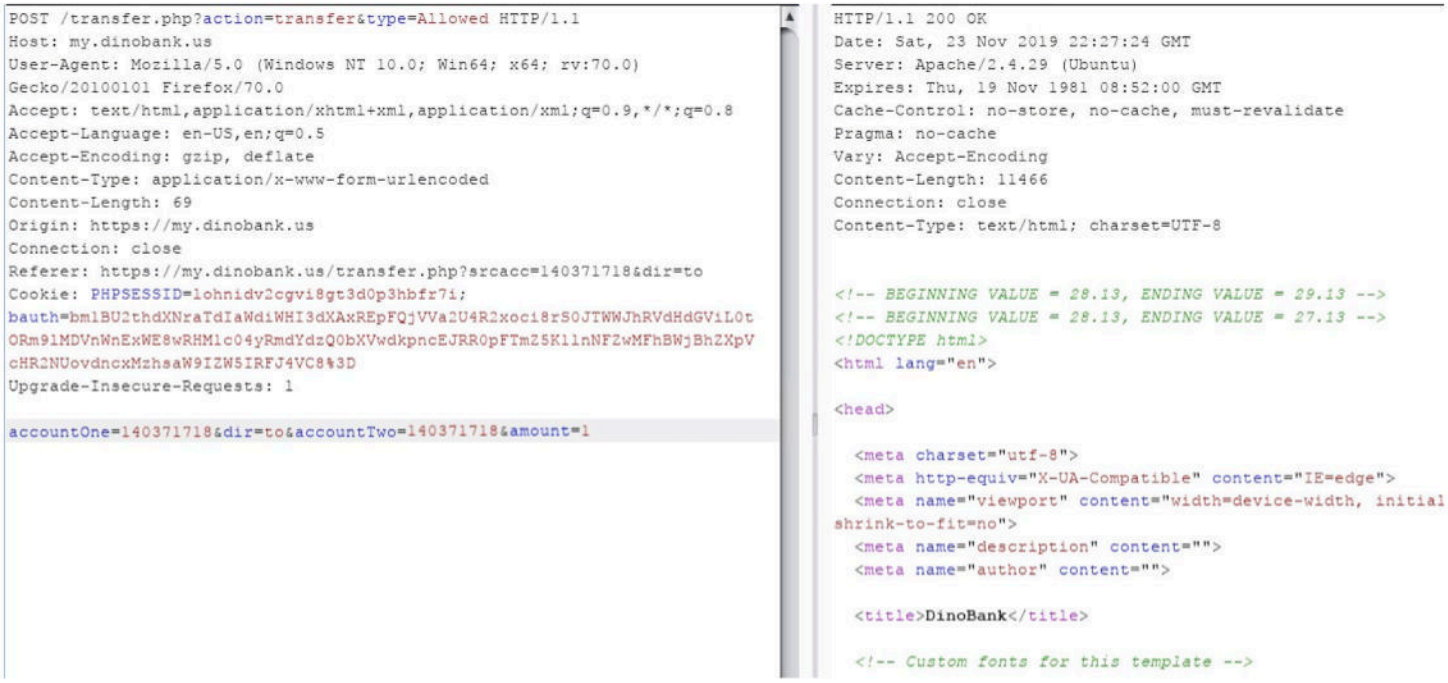
Threat Level: **High (8.2)**

Description:
DinoBank allows customers to transfer money electronically via the DinoBank website. Due to errors in validating transaction amounts, it was observed that a malicious user could initiate transactions that led to gaining unbounded amounts of money in their account. Note that this vulnerability was originally included in the initial penetration test report and remains unremediated.

Potential Business Impact:
As a financial institution, manipulation of account balances could lead to large financial losses and massive loss of customer trust for DinoBank, as well as potential regulatory violations.

Affected Host:
10.0.2.101 -- BANKWEB-01 (my.dinobank.us)

Exploitation Details:
The “transfer.php” endpoint on the core DinoBank banking services can be exploited to gain infinite account balances. It was observed that upon initiating transfers from an account to itself, the account’s balance would not be deducted the proper amount and would only increase, as shown in the screenshot below. As a result, this can be exploited to obtain an infinite account balance.



```
POST /transfer.php?action=transfer&type=Allowed HTTP/1.1
Host: my.dinobank.us
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:70.0)
Gecko/20100101 Firefox/70.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 69
Origin: https://my.dinobank.us
Connection: close
Referer: https://my.dinobank.us/transfer.php?srcacc=140371718&dir=to
Cookie: PHPSESSID=lohnidv2cgvi8gt3d0p3hbfr7i;
bauth=bmlBU2thdXNraTdIaWdiWHI3dXAxREpFQjVVa2U4R2xoci8rS0JTWWJhRVdHdGViL0t
ORm9lMDVnWnExWE8wRHMlc04yRmdYdzQ0bXVwdkpncEJRR0pFTmZ5KllnNFZwMFhBWjBhZXpV
cHR2NUovdncxMzhsaW9IZW5IRFJ4VC8%3D
Upgrade-Insecure-Requests: 1

accountOne=140371718&dir=to&accountTwo=140371718&amount=1
```

```
HTTP/1.1 200 OK
Date: Sat, 23 Nov 2019 22:27:24 GMT
Server: Apache/2.4.29 (Ubuntu)
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate
Pragma: no-cache
Vary: Accept-Encoding
Content-Length: 11466
Connection: close
Content-Type: text/html; charset=UTF-8


<!-- BEGINNING VALUE = 28.13, ENDING VALUE = 29.13 -->
<!-- BEGINNING VALUE = 28.13, ENDING VALUE = 27.13 -->
<!DOCTYPE html>
<html lang="en">

<head>

  <meta charset="utf-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial
shrink-to-fit=no">
  <meta name="description" content="">
  <meta name="author" content="">

  <title>DinoBank</title>

  <!-- Custom fonts for this template -->
```

Transferring from an account to itself leads to an increase in the account's balance

Remediation Recommendations:

Ensure that an account may not transfer money to itself. Any such currency transactions or suspicious activity, even if they were able to take place after remediation and instatement of a fraud-monitoring system, should be audited and immediately reported as per Bank Secrecy Act requirements.

Exposure of Customer Account Information (MOU: PSWD, CDATA)

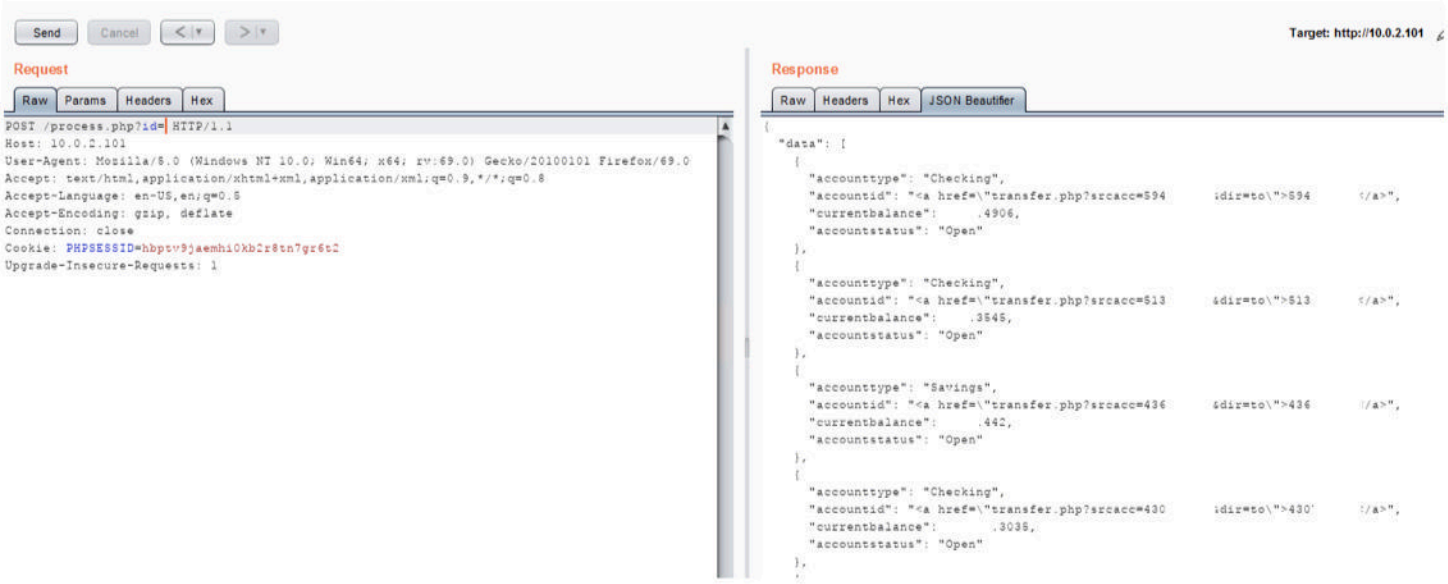
Threat Level: **High (8.2)**

Description:
The DinoBank core website contains a hidden page that returns customer information, including account numbers and balances. This website is accessible without authentication, exposing sensitive customer information to any website visitor. Note that this vulnerability was originally included in the initial penetration test report and remains unremediated.

Potential Business Impact:
Exposed account numbers and balances can lead to both financial and privacy loss. For instance, an unauthenticated attacker may abuse the fact that account transfers are permitted from any bank account in order to steal money from all listed accounts. Furthermore, accessing customer balance information is an infringement of privacy and potentially a violation of DinoBank's legally-required Privacy Policy.

Affected Host:
10.0.2.101 -- BANKWEB-01 (my.dinobank.us)

Exploitation Details:
Visiting http://10.0.2.101/process.php while supplying any value for the id parameter returns customer account ids, as pictured below. Note that this endpoint is accessible without authentication, leading to information disclosure to any public visitor.



Exposed customer account ids and balances

Remediation Recommendations:
Enforce access control on this page so that users may only view account information they are entitled to.

References:
Access Control: Principles and Practice

Internal API Server Accessible Via Hardcoded API Keys (MOU: CDATA)

Threat Level: High (8.2)

Description:
The public DinoBank website relies on an intermediate API server to process backend database queries. It was observed that access to this intermediate server is not restricted to the production server, and authentication can be provided via a static, hardcoded API key. As a result, an attacker can directly query this API service, accessing bulk customer information and executing transactions at will.

Note that a similar vulnerability was originally included in the initial penetration test report, though the API key has been rotated.

Potential Business Impact:
This results in bulk PII exposure of over 20,000 DinoBank customers, including names, taxpayer IDs (Social Security numbers or employer identification numbers), phone numbers, email addresses, home addresses, and bank PINs. Other accessible endpoints expose account, transaction, loan, and Certificate of Deposit information. Furthermore, an attacker may modify database contents by making crafted requests, resulting in forged transactions or account information. Of course, such a bulk breach would violate not only DinoBank's MOU, but a plethora of applicable regulations.

Affected Host:
10.0.2.100 -- CORE-01

Exploitation Details:
Using an API key obtained from the DinoBank core webserver, our security engineers were able to make authenticated requests to the API service at http://10.0.2.100. This included endpoints such as "/v1/customers", which returned detailed PII of all DinoBank customers.

Accessing customer account information

Other accessible endpoints included "/v1/accounts/", "/v1/utils/", and "/v1/txs/", and "/v1/employees", all of which exposed sensitive information.

Furthermore, by making PUT requests to certain database objects, such as "/v1/accounts/customer_id", it is possible to modify customer information, although our engineers refrained from modifying any production data.

Remediation Recommendations:
Enforce network access control to ensure that the API server is only accessible from the production server, and that authentication is performed using industry best practices such as public key authentication.

References:
Access Control: Principles and Practice

Missing Account Validation for Money Transfers (MOU: CDATA, AUDIT)

Threat Level: High (7.5)

Description:
The DinoBank website fails to perform any form of verification when initiating account transfers. As a result, given only an account ID, a malicious user may transfer any amount of money from that account with no direct interaction. Note that this vulnerability was originally included in the initial penetration test report and remains unremediated.

Potential Business Impact:
Malicious users have the ability to transfer money from other customers' accounts, allowing monetary theft from an arbitrary user's account. Any theft performed via this vulnerability would compromise the integrity of DinoBank's data as well as severely degrading users' trust in the corporation.

Affected Host:
10.0.2.101 -- BANKWEB-01 (my.dinobank.us)

Exploitation Details:

Observe that upon making a request to "/transfer.php", any arbitrary AccountID can be specified for the to/from address. Submitting another user's account ID, such as exposed by the process.php endpoint, results in monetary theft. (See "Exposure of Customer Account Information" finding for details.)

Remediation Recommendations:
Restrict access so that users are only able to send from accounts that the user owns. Also, as previously mentioned, audit and report all suspicious or large-volume transfers to both DinoBank security and compliance staff and Federal, state and local entities as required by KYC/AML and CFT laws.

Passwords Stored in Plaintext (MOU: PSWD)

Threat Level: **High (7.5)**

Description:
In Windows machines, credentials for QueryTree, MediaWiki, and DinoBank are stored in plaintext, with no hashing or salting. Furthermore, a database served on a DinoBank core infrastructure system also stores user passwords in plaintext. This places personnel at significant risk in the case where the Windows machines or the database are breached and passwords stolen. It also places DinoBank at significant risk of non-compliance with key regulations like PCI-DSS and FDIC rules.

Potential Business Impact:
This could result in large legal penalties for not properly storing confidential data, in addition to reputational damage, loss of consumer trust and associated loss of profits. Under PCI-DSS item 8.4 and as a requirement for FDIC insurance, both of which are essential to DinoBank, plaintext passwords are forbidden.

Affected Hosts:
10.0.2.100 -- CORE-01
10.0.10.201 -- GOTHAM-TLR-01
10.0.10.202 -- GOTHAM-TLR-02
10.0.10.203 -- GOTHAM-TLR-03
10.0.10.208 -- GOTHAM-WK-01
10.0.10.209 -- GOTHAM-WK-02
10.0.11.201 -- METRO-TLR-01
10.0.11.202 -- GOTHAM-WK-02
10.0.11.208 -- GOTHAM-WK-03
10.0.12.201 -- SPRING-TLR-01
10.0.12.208 -- SPRING-WK-01

Exploitation Details:
The Windows machines hosted on 10.0.10.201, 10.0.10.202, 10.0.10.203, 10.0.10.208, 10.0.10.209. 10.0.11.201, 10.0.11.202, 10.0.11.208, 10.0.12.201, 10.0.12.208 possessed files storing passwords in plaintext.

Passwords file in Windows machine 10.0.11.201

To obtain the passwords, we utilized remote desktop to access the Windows machines. The files were placed in the Desktop folder.

Moreover, the PostgreSQL database hosted on 10.0.2.100 stores passwords of over 8,000 users in plaintext. All the attacker must do to gain access to user passwords is dump the contents of the database, with no further effort required.

Remediation Recommendations:
Passwords should be hashed and salted for secure storage under a slow hashing algorithm. Certain hashing algorithms such as bcrypt take much more time and computer power to crack, and are significantly better than plaintext. However, to mitigate the risk from rainbow tables or dictionary attacks, the hashed passwords should also be salted with random data to further add complexity and time to the cracking process. Furthermore, passwords should never be stored in plaintext files on a system, rather a password manager enforcing strong encryption, random generation and multi-factor authentication should be used.

References:
John Wu, *Stack Overflow*
Auth0, *Adding Salt to Hashing*

Internal API Path Traversal via Account ID (MOU: CDATA)

Threat Level: **High (7.3)**

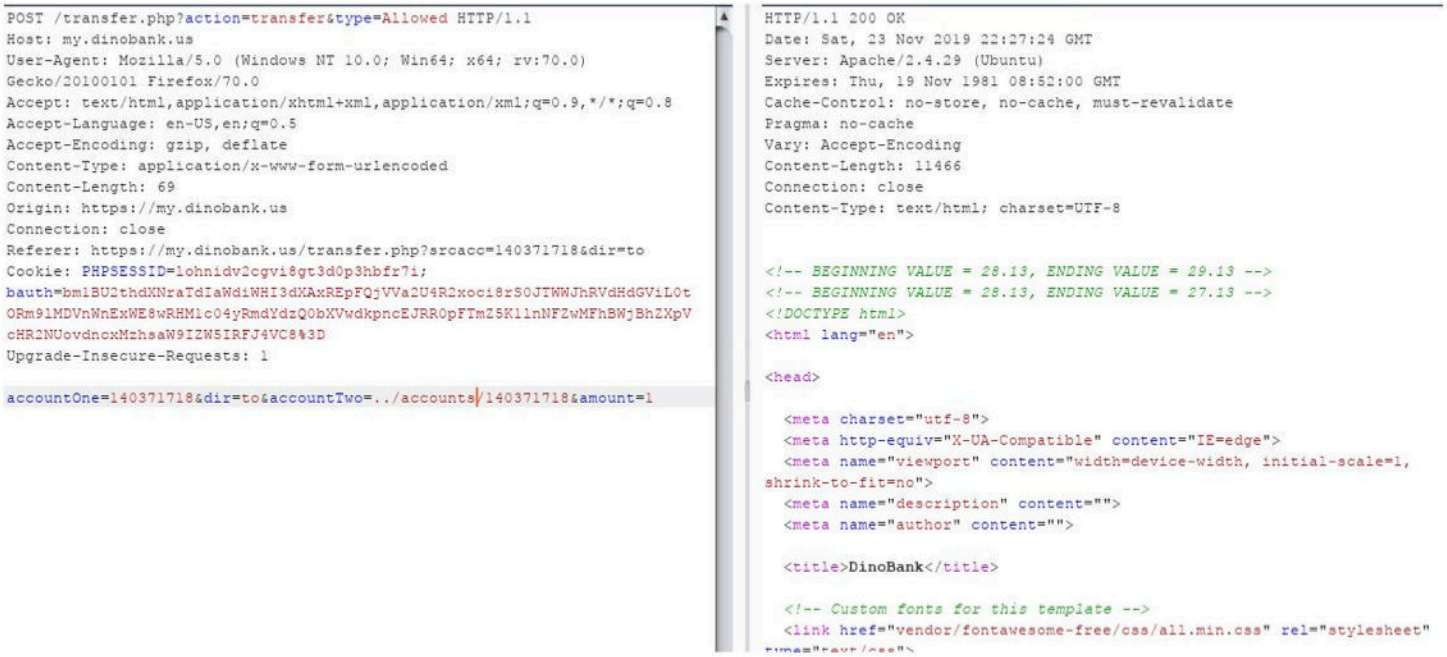
Description:
It was observed that several endpoints of the core DinoBank webserver are vulnerable to path traversal via account ID parameters in requests to an internal API. As a result, an attacker can make authenticated requests to arbitrary API endpoints on the internal DinoBank API.

Potential Business Impact:
An attacker may exploit this vulnerability to expose or modify database information by making arbitrary crafted requests to the DinoBank internal API.

Affected Host:
10.0.2.101 -- BANKWEB-01 (my.dinobank.us)

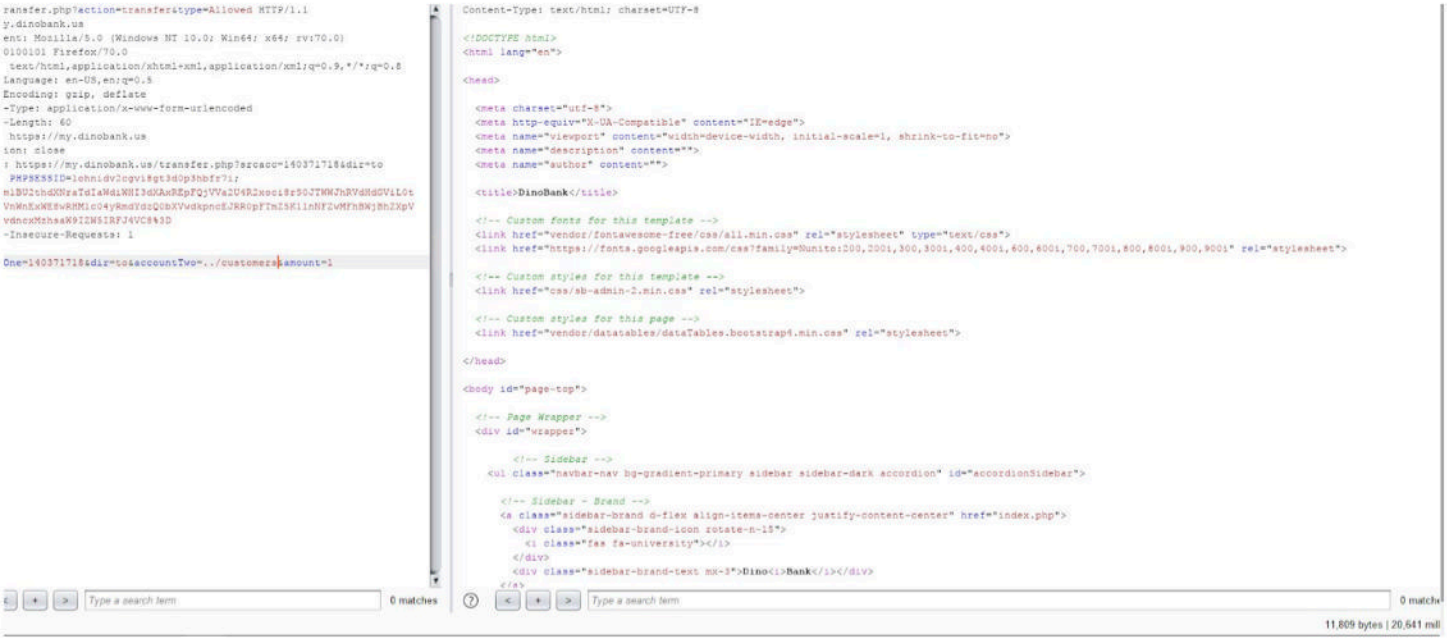
Exploitation Details:
Visit the DinoBank transfer page at https://my.dinobank.us/transfer.php.

After sending a valid transaction, prepend one account ID parameter with the value "../accounts/". This has the effect of stepping one directory up in the request to the internal API, and again accessing the accounts endpoint. The transaction will succeed as the same endpoint is being called.



Modifying the account ID to traverse API paths

Next, change the account id to "../customers". Observe that the request will take around 20 seconds to complete, as the complete customer database is being loaded.



Traversing to /v1/customers, which takes over 20 seconds to query

Remediation Recommendations:

Escape all special characters present in user input to prevent path traversal in intermediate API requests.

Medium Risk

Cross-Site Request Forgery on Core Banking Actions (MOU: CDATA)

Threat Level: Medium (6.5)

Description:
It was observed that several critical actions in the DinoBank website did not enforce Cross-Site Request Forgery (CSRF) protection. As a result, an attacker can force customers to take certain actions, such as deleting accounts or transferring money with minimal user interaction.

Note that this vulnerability was originally included in the initial penetration test report and remains unremediated.

Potential Business Impact:
An attacker may exploit this vulnerability to target individual customer accounts, leading to forced transfers or deletion of DinoBank accounts.

Affected Host:
10.0.2.101 -- BANKWEB-01 (my.dinobank.us)

Exploitation Details:

Observe that while taking actions in DinoBank, such as deleting accounts, transferring money, or creating new accounts, no CSRF protection is applied. In the case of a GET request, this can be verified by visiting the URL to perform the transaction. In the case of a POST request, this can be verified by creating an HTML form with the contents of the request.

Affected endpoints:

- GET new.php: used for creating new banking accounts
- POST close.php: used for deleting customer accounts
- GET transfer.php: used for transferring funds between accounts

```
POST /close.php?action=d HTTP/1.1
Host: 10.0.2.101
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:69.0) Gecko/20100101
Firefox/69.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 0
Connection: close
Referer: http://10.0.2.101/close.php
Cookie: PHPSESSID=22kiuigtotm5a5anu597l3q0q6;
bauth=ZTBsWVZuQm1DMTNUejF3OUNvN2h5d0VLaFErcVh1Vi9KaFJyNGM5ZXoyam5qSn1FaWJxZ2VOb0V1N
TBUWTh2SU9jd1RNdWxlV1R0WGJNRjdpRFJic0E9PQ%3D%3D
Upgrade-Insecure-Requests: 1
```

The request to delete a customer's account, which includes no CSRF protection

Remediation Recommendations:

Enforce CSRF protection for all sensitive account actions.

References:
CSRF protection cheat sheet

Unauthenticated FTP Server (MOU: PSWD)

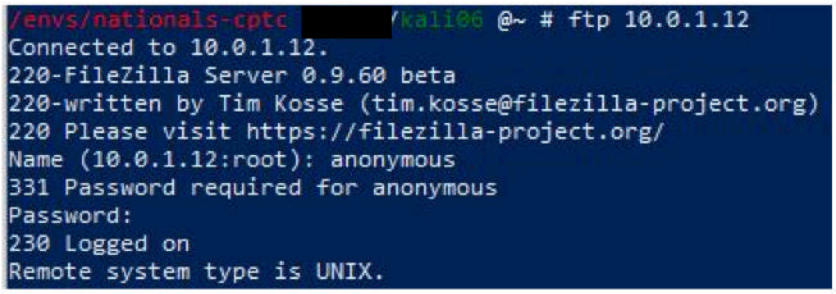
Threat Level: Medium (5.3)

Description:
There exists a file-sharing server with anonymous, unauthenticated login.

Potential Business Impact:
This security vulnerability exposes the full filesystem of a file server to read-only remote access. Unauthenticated users are able to download any file on the server for offline analysis using the default login. As this server is responsible for providing Windows updates to other hosts in the network, an attacker could perform an offline search for misconfigurations that would allow interference with the Windows Update service.

Affected Host:
10.0.1.12 -- CORP-WSUS-01

Exploitation Details:
Connecting with an FTP client to 10.0.1.12 and authenticating with username “anonymous” and no password allowed interaction with the filesystem of the server, as shown below. All files on the server could be downloaded locally with the “get” command, though write access was restricted. Visible data included application configuration files and user command history. This vulnerability was initially found in our previous engagement.



A directory listing of C:\ through an anonymous FTP connection, requiring no password

Remediation Recommendations:
Disable all anonymous logins to exposed file servers. For additional security, only allow encrypted connections via SFTP.

References:
Securing a FileZilla FTP Server

Reflected Cross-Site Scripting (MOU: CDATA)

Threat Level: Medium (5.3)

Description:
Multiple reflected Cross-Site Scripting vulnerabilities exist on the DinoBank website, allowing an attacker to execute JavaScript in the browser of DinoBank customers.

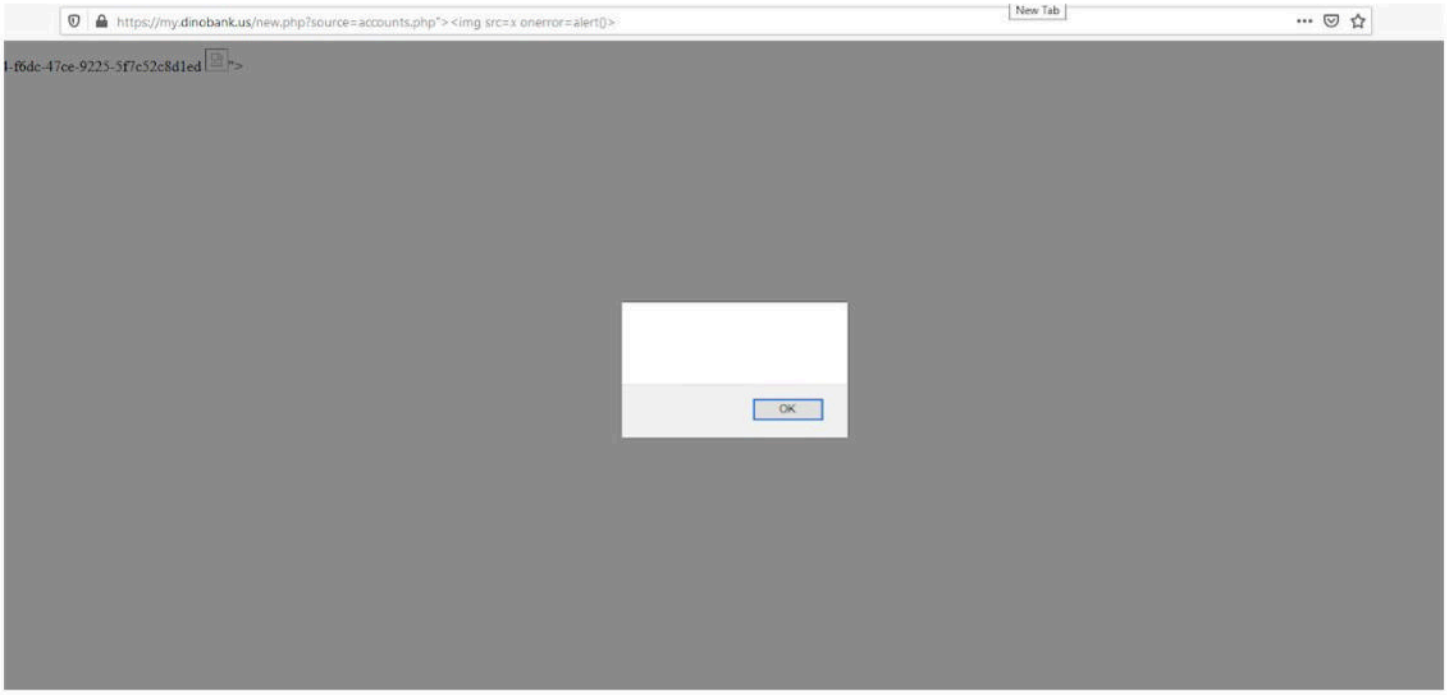
Potential Business Impact:
As with standard reflected Cross-Site-Scripting vulnerabilities, an attacker can compromise customer accounts by executing JavaScript to steal session cookies or take malicious actions. This can lead to loss of private information and account money.

Affected Host:
10.0.2.101 -- BANKWEB-01 (my.dinobank.us)

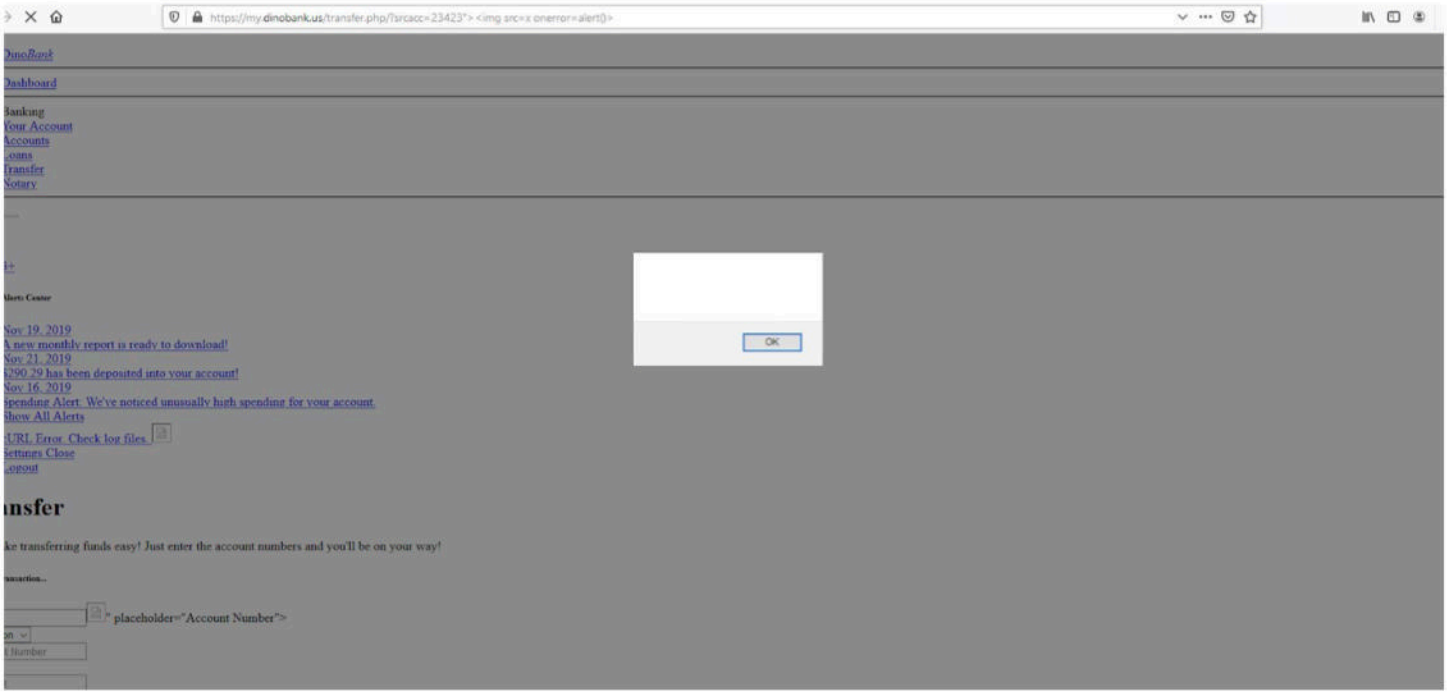
Exploitation Details:

It was observed that visiting the following pages allows execution of arbitrary JavaScript, as demonstrated by the execution of a JavaScript alert message:

- http://10.0.2.101/new.php?source=accounts.php%22%3E%3Cimg%20src=x%20onerror=alert(document.domain)%3E
- http://10.0.2.101/transfer.php?srcacc=594583946%22%3E%3Cimg%20src=x%20onerror=alert()%3E&dir=to

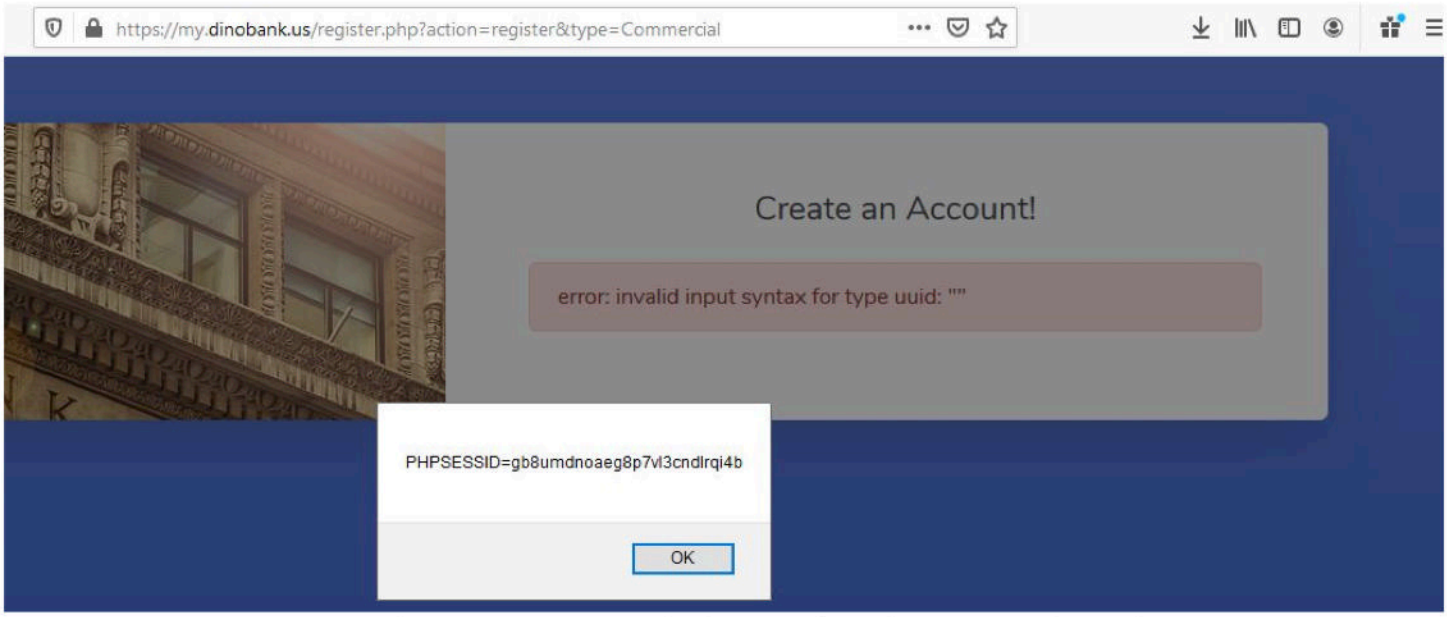


Executing arbitrary JavaScript in the browser from the new.php endpoint



Executing arbitrary JavaScript in the browser from the transfer.php endpoint

In addition, during this round of testing our engineers found an additional reflected Cross-Site Scripting vulnerability in the register.php endpoint of the core DinoBank web server. Specifically, it was observed that entering JavaScript into the password field allows execution of arbitrary JavaScript, as demonstrated by the execution of a JavaScript alert message:



Executing arbitrary JavaScript in the browser from the register.php endpoint

Remediation Recommendations:

For the referenced pages, ensure that all user input is properly escaped before being rendered. As a best practice, front-end display should be standardized using a framework such as Angular or ReactJS in order to systematically prevent these vulnerabilities.

References:
Preventing XSS in PHP

Open Access to ATM Operator Menu (MOU: PSWD, CDATA, AUDIT)

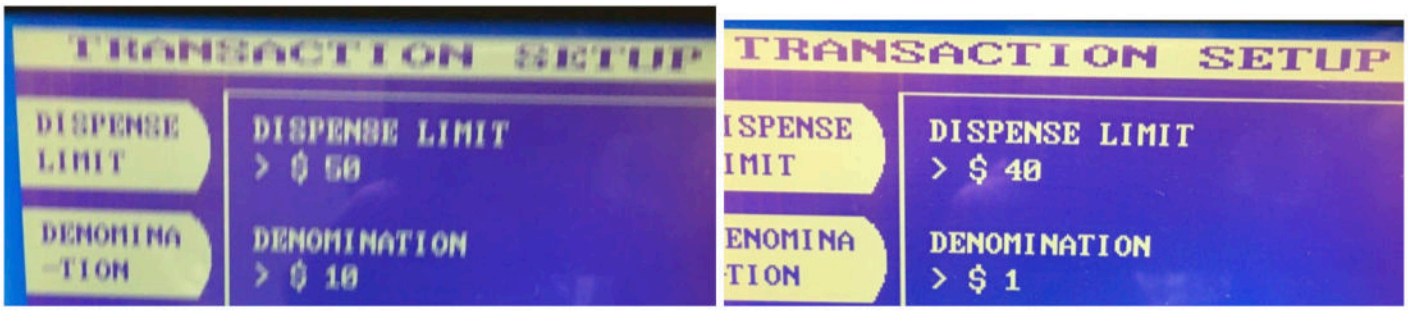
Threat Level: Medium (5.3)

Description:
The ATM operator menu was obtained via a screen left open by a DinoBank ATM support engineer.

Potential Business Impact:
Any changes to the ATMs which leave them in a state that should not be seen by the client is a dangerous situation which could potentially put the machine at risk of being compromised.

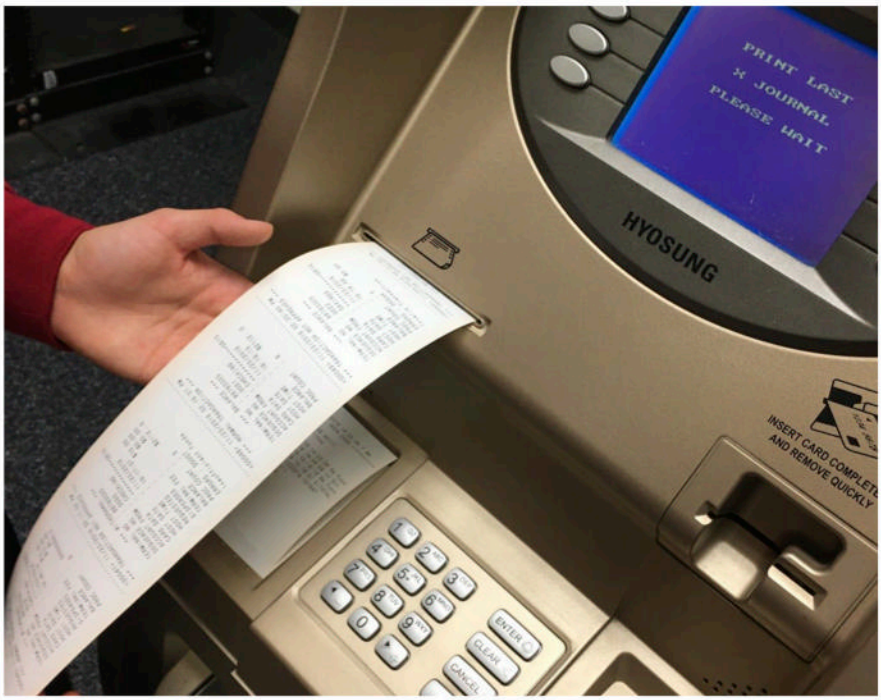
Affected Host:
Physical ATM (Hyosung 1500)

Exploitation Details:
At around 12:30 pm, a DinoBank support engineer entered the engagement's testing room and approached the ATM. Once they exited the premises, our engineers observed that the support engineer had entered the operator code into the ATM and left the operator menu unattended. From there, our engineers were able to perform multiple attacks on the ATM's functionality, including changing the denomination of notes held by the cassette, enabling the ATM to charge a surcharge on all transactions, printing all transactions previously performed that day, and exfiltrating ATM version and setup information, all of which is shown respectively in the screenshots below.

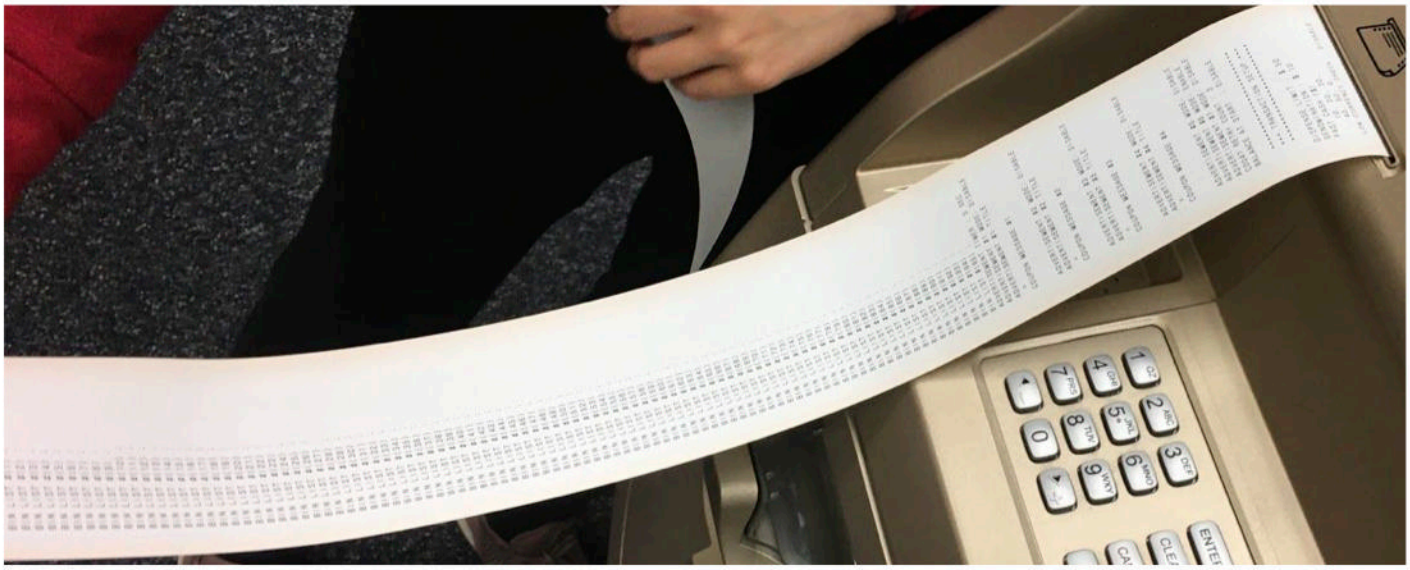


Pair of images showing default denomination ($10) and changed denomination ($1)

Pair of images showing surcharge mode changing from disabled to enabled



Printing a list of transactions made that day



Printing ATM version and configuration information

Note that our engineers reverted the changes made above wherever possible.

Remediation Recommendations:
To ensure all staff are clear on the policy relating to ATM security, we recommend a standardized training where security best practices are taught and practiced. When support engineers change settings/tend to the ATMs, they must be careful to close all open sessions and return the ATM back to its original state as usually seen by the customer. Furthermore, any and all ATM privileged-access attempts must be audited.

Private RSA Key Exposed in Public GitHub Repository (CDATA: SECGOV, AUDIT)

Threat Level: Medium (4.3)

Description:
A private RSA key was discovered to have been inadvertently committed to a DinoBank employee's GitHub account. One of our security consultants noticed this issue beforehand and attempted to contact the affected employee, though no response was received (see discussion on implementing a vulnerability disclosure policy in the Introduction).

Note that this key was originally included in the initial penetration test report and remains public.

Potential Business Impact:
Although no hosts were discovered that allowed connection via this RSA key, it is possible that other development servers exist that rely on this key. If such servers do exist, an attacker may compromise such servers and pivot to other internal networks.

Affected Host:
N/A - This was the result of open-source reconnaissance conducted in preparation for this penetration test.

Exploitation Details:

The GitHub user DinoDanOliver committed a private RSA key, accessible at the following github: https://github.com/DinoDanOliver/.files/commit/cb765593bd416c9f573f21ca1c1b07bdccfe14d6.
While the user has since deleted the key, it is still accessible via GitHub commit history.

Remediation Recommendations:
The private key in question should be removed from GitHub and rotated from all servers using the key. Furthermore, employees should be instructed to use only official GitHub accounts for storing company information. The use of automated audit tools to search for inadvertently posted keys on GitHub repositories may assist.

Default Bank PIN Codes (MOU: PSWD, CDATA)

Threat Level: Medium (4.3)

Description:
PIN codes on IVR for all users in the postgreSQL database are 0000.

Potential Business Impact:
Although it did not pose an immediate risk in the environment our engineers observed, user PINs should not stay default much longer after they are initialized. If sensitive information can be accessed by the PINs sometime in the future, then the users are all put at immediate risk.

Affected Host:
10.0.1.102 -- IVR-01, in connection with the provided IVR system

Exploitation Details:
From a previous vulnerability in the postgreSQL database, our engineers identified that every customer of DinoBank still maintained the default PIN (specifically, 0000) for accessing their account information on the IVR system. Our engineers were able to confirm this by calling the IVR system with that customer's tax ID and authenticating with the default PIN. This allowed our engineers to access sensitive loan, Certificate of Deposit, and account balance information of a customer, which is a breach of user data and a potential violation of the Gramm-Leach-Bliley Act.

Remediation Recommendations:
Ensure that all users change their PINs as soon as possible to a code other than the default 0000. More generally, ensure that all systems which have a default password or PIN are switched from the default as soon as possible.

Intranet Wiki Edit Access and Information Leakage (MOU: CDATA)

Threat Level: Medium (4.3)

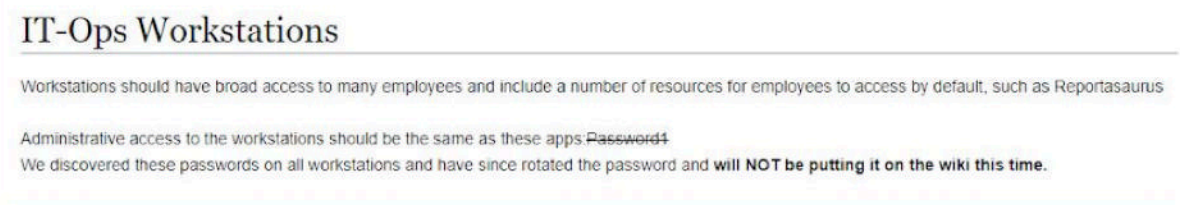
Description:
The DinoBank Intranet Wiki contains multiple references to credentials in plaintext. Additionally, the Wiki allows anyone, including users without an account, to “Edit” and “Edit Source” on multiple pages, which allows for a possible watering hole attack in the event of compromise or an insider threat.

Potential Business Impact:
Plaintext credentials provide further information to potential attackers conducting reconnaissance. Furthermore, by editing wiki pages, an attacker may misinform DinoBank employees of company policies, causing undesired actions.

Affected Host:
10.0.1.31 -- WEB-01

Exploitation Details:

Visiting http://10.0.1.31/index.php?title=IT-Ops_Workstations, observe that default credentials are listed in the Wiki page:



Previous administrative password listed in plaintext on the Intranet Wiki

Furthermore, under the network access page, additional default passwords are listed:

An initial network password listed in plaintext on the Intranet Wiki

Lastly, an attacker may edit any Wiki page in order to spread misinformation to DinoBank employees.

Remediation Recommendations:
Employees should be given edit access on an as-needed basis, per the Principle of Least Privilege. In addition, a review of access controls for all pages that allow editing is highly recommended, as this vulnerability is

similar to a vulnerability reported in the previous engagement, in which edit access was allowed for any public user to a public-facing bakery website.

Finally, plaintext passwords, even “old” passwords, should be avoided from being published, as not all users may have changed their passwords yet and additionally users often tend to select new passwords based on previous passwords. For example, as referenced in the Weak Password for Local Admin Credentials finding earlier, the publication of an old password allowed our security engineers to intelligently guess the new password being used.

Low Risk

Information Disclosure - Listable Directories

Threat Level: Low (3.3)

Description:
Several web servers are configured to allow directory listing. This vulnerability was also found in our previous engagement.

Potential Business Impact:
Attackers can enumerate files, directories, and information about the application and host server that would otherwise be inaccessible. If sensitive internal information about the file system is exposed, attackers could utilize this critical data to compromise the application or server.

Affected Hosts:
10.0.2.101 -- BANKWEB-01 (my.dinobank.us)

Exploitation Details:
A tool called Directory Buster was used to enumerate the directories that had listing enabled. Our security engineers could additionally confirm that directory listing was enabled for the specified paths by visiting the directories in their browser.

```
---- Entering directory: http://10.0.2.101/css/ ----
(!) WARNING: Directory IS LISTABLE. No need to scan it.
    (Use mode '-w' if you want to scan it anyway)

---- Entering directory: http://10.0.2.101/img/ ----
(!) WARNING: Directory IS LISTABLE. No need to scan it.
    (Use mode '-w' if you want to scan it anyway)
==> DIRECTORY: http://10.0.2.101/img/user/

---- Entering directory: http://10.0.2.101/js/ ----
(!) WARNING: Directory IS LISTABLE. No need to scan it.
    (Use mode '-w' if you want to scan it anyway)
==> DIRECTORY: http://10.0.2.101/js/demo/

---- Entering directory: http://10.0.2.101/img/user/ ----
(!) WARNING: Directory IS LISTABLE. No need to scan it.
    (Use mode '-w' if you want to scan it anyway)

---- Entering directory: http://10.0.2.101/js/demo/ ----
(!) WARNING: Directory IS LISTABLE. No need to scan it.
    (Use mode '-w' if you want to scan it anyway)
```

Excerpt of output from a Directory Buster scan on 10.0.2.101

Remediation Recommendations:
Turn off directory listing for all referenced web servers in order to prevent the exposure of potentially sensitive file system information.

References:
Information Exposure Through Directory Listing
Disable Directory Listing on Web Servers

Exposure of Internal Server Information

Threat Level: Low (3.9)

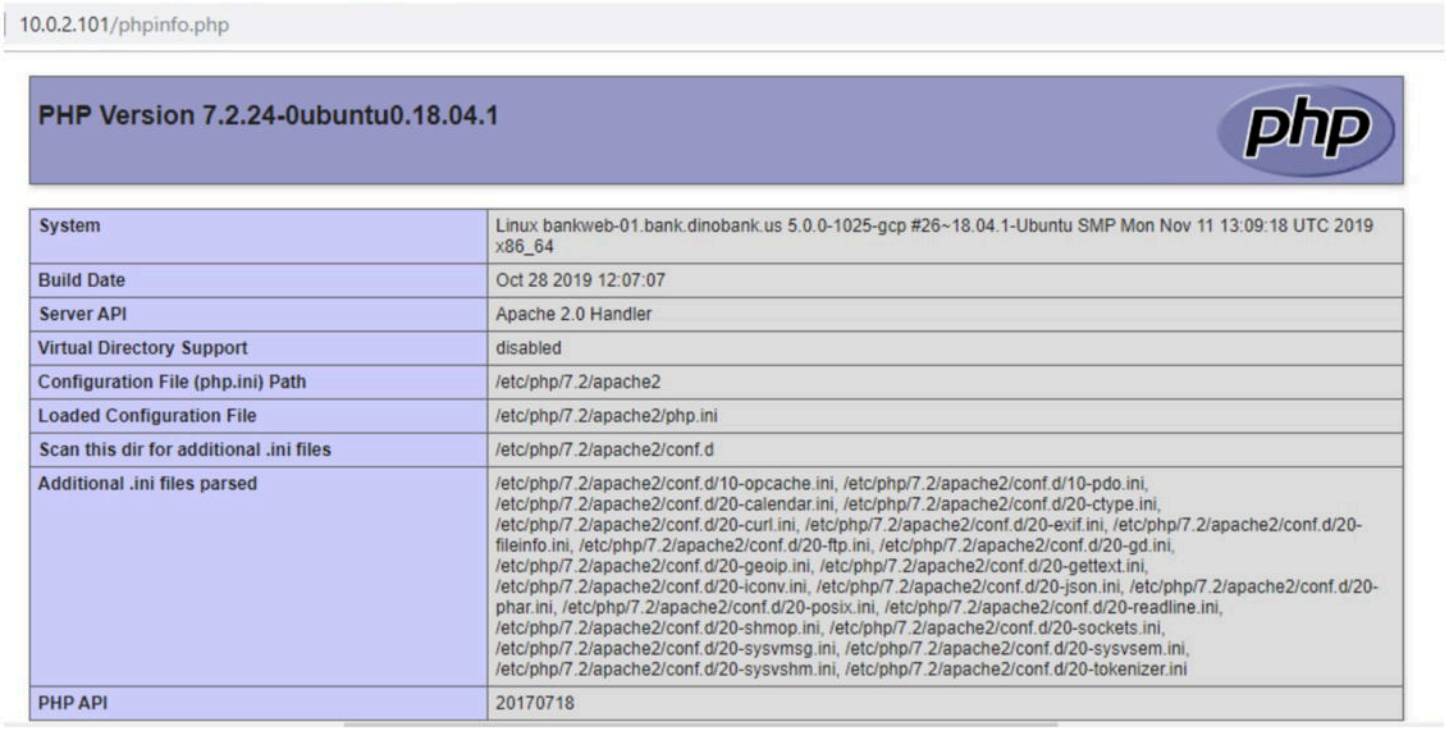
Description:
It was observed that sensitive internal information of the DinoBank website was returned through PHP dumps of API responses.

Potential Business Impact:
Exposure of such information can aid an attacker in targeting the DinoBank website and expose sensitive internal configurations.

Affected Host:
10.0.2.101 -- BANKWEB-01 (my.dinobank.us)

Exploitation Details:

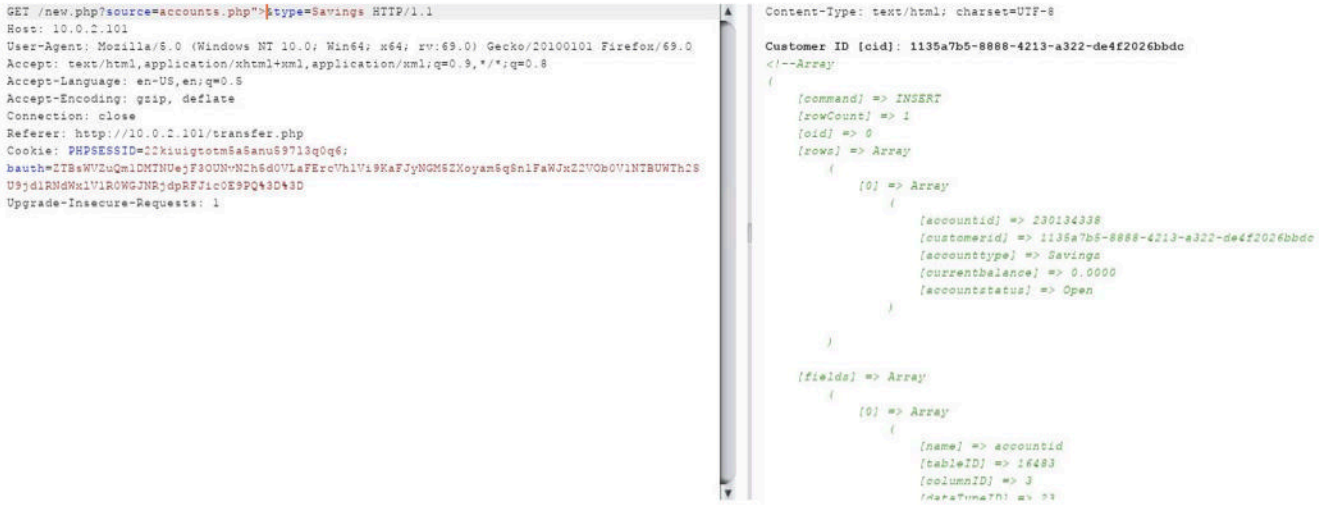
Upon visiting https://my.dinobank.us/phpinfo.php, observe that the full phpinfo() page is returned. This includes PHP server information and server environment variables.



System	Linux bankweb-01.bank.dinobank.us 5.0.0-1025-gcp #26~18.04.1-Ubuntu SMP Mon Nov 11 13:09:18 UTC 2019 x86_64
Build Date	Oct 28 2019 12:07:07
Server API	Apache 2.0 Handler
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc/php/7.2/apache2
Loaded Configuration File	/etc/php/7.2/apache2/php.ini
Scan this dir for additional .ini files	/etc/php/7.2/apache2/conf.d
Additional .ini files parsed	/etc/php/7.2/apache2/conf.d/10-opcache.ini, /etc/php/7.2/apache2/conf.d/10-pdo.ini, /etc/php/7.2/apache2/conf.d/20-calendar.ini, /etc/php/7.2/apache2/conf.d/20-ctype.ini, /etc/php/7.2/apache2/conf.d/20-curl.ini, /etc/php/7.2/apache2/conf.d/20-exif.ini, /etc/php/7.2/apache2/conf.d/20-fileinfo.ini, /etc/php/7.2/apache2/conf.d/20-ftp.ini, /etc/php/7.2/apache2/conf.d/20-gd.ini, /etc/php/7.2/apache2/conf.d/20-geoip.ini, /etc/php/7.2/apache2/conf.d/20-gettext.ini, /etc/php/7.2/apache2/conf.d/20-iconv.ini, /etc/php/7.2/apache2/conf.d/20-json.ini, /etc/php/7.2/apache2/conf.d/20-phar.ini, /etc/php/7.2/apache2/conf.d/20-posix.ini, /etc/php/7.2/apache2/conf.d/20-readline.ini, /etc/php/7.2/apache2/conf.d/20-shmop.ini, /etc/php/7.2/apache2/conf.d/20-sockets.ini, /etc/php/7.2/apache2/conf.d/20-sysvmsg.ini, /etc/php/7.2/apache2/conf.d/20-sysvsem.ini, /etc/php/7.2/apache2/conf.d/20-sysvshm.ini, /etc/php/7.2/apache2/conf.d/20-tokenizer.ini
PHP API	20170718

phpinfo() page returned

Furthermore, it was observed that upon making requests to endpoints such as accounts.php, raw PHP dumps of internal API responses would be returned:



PHP dump in API response

Remediation Recommendations:

Disable SQL errors and refrain from dumping internal PHP objects.

SMB Enumeration

Threat Level: Low (3.1)

Description:
Guest users with no password were able to enumerate SMB shares and named pipes. Furthermore, these unauthenticated users could list the IPC$ share.

Potential Business Impact:
Exposure of this information can aid an attacker in taking control of the system or gaining information stored in the Windows Domain Controllers.

Affected Hosts:
10.0.10.100 – GOTHAM-DC
10.0.11.100 – METRO-DC
10.0.12.100 – SPRING-DC

Exploitation Details:
Our security engineers utilized SMBMap in order to enumerate these shares. After specifying the host, username “guest”, and password “”, they were able to list all shares and named pipes.

```
envs/nationals-cptc [redacted] kali01 @~ # smbmap -H 10.0.11.201 -u Administrator -p Password2 -s ADMIN$
+] Finding open SMB ports....
+] User SMB session established on 10.0.11.201...
+] IP: 10.0.11.201:445 Name: nationals-t7-branch-metro-metro-tlr-01.c.infra-test-environment.internal
        Disk                                                    Permissions     Comment
        ----                                                    -----------     -------
        ADMIN$                                                  READ, WRITE     Remote Admin
        C$                                                      READ, WRITE     Default share
        .
        fr--r--r--                3 Mon Jan  1 00:00:00 1601    InitShutdown
        fr--r--r--                4 Mon Jan  1 00:00:00 1601    lsass
        fr--r--r--                3 Mon Jan  1 00:00:00 1601    ntsvcs
        fr--r--r--                3 Mon Jan  1 00:00:00 1601    scerpc
        fr--r--r--                1 Mon Jan  1 00:00:00 1601    Winsock2\CatalogChangeListener-270-0
        fr--r--r--                1 Mon Jan  1 00:00:00 1601    Winsock2\CatalogChangeListener-188-0
        fr--r--r--                3 Mon Jan  1 00:00:00 1601    epmapper
        fr--r--r--                3 Mon Jan  1 00:00:00 1601    LSM_API_service
        fr--r--r--                3 Mon Jan  1 00:00:00 1601    eventlog
        fr--r--r--                1 Mon Jan  1 00:00:00 1601    Winsock2\CatalogChangeListener-390-0
        fr--r--r--                3 Mon Jan  1 00:00:00 1601    atsvc
        fr--r--r--                3 Mon Jan  1 00:00:00 1601    TermSrv_API_service
        fr--r--r--                3 Mon Jan  1 00:00:00 1601    Ctx_WinStation_API_service
        fr--r--r--                1 Mon Jan  1 00:00:00 1601    Winsock2\CatalogChangeListener-368-0
        fr--r--r--                5 Mon Jan  1 00:00:00 1601    wkssvc
        fr--r--r--                3 Mon Jan  1 00:00:00 1601    SessEnvPublicRpc
        fr--r--r--                1 Mon Jan  1 00:00:00 1601    Winsock2\CatalogChangeListener-204-0
        fr--r--r--                3 Mon Jan  1 00:00:00 1601    spoolss
        fr--r--r--                3 Mon Jan  1 00:00:00 1601    trkwks
        fr--r--r--                1 Mon Jan  1 00:00:00 1601    Winsock2\CatalogChangeListener-6ec-0
        fr--r--r--                4 Mon Jan  1 00:00:00 1601    srvsvc
        fr--r--r--                1 Mon Jan  1 00:00:00 1601    Winsock2\CatalogChangeListener-648-0
        fr--r--r--                1 Mon Jan  1 00:00:00 1601    pyc-2124-0-ku_7pe91
        fr--r--r--                3 Mon Jan  1 00:00:00 1601    W32TIME_ALT
        fr--r--r--                1 Mon Jan  1 00:00:00 1601    PIPE_EVENTROOT\CIMV2SCM EVENT PROVIDER
        fr--r--r--                1 Mon Jan  1 00:00:00 1601    Winsock2\CatalogChangeListener-1fc-0
        fr--r--r--                1 Mon Jan  1 00:00:00 1601    splunk-powershell-pipe-9c8ef5c511458dfa
        fr--r--r--                1 Mon Jan  1 00:00:00 1601    PSHost.132189879690163391.3252.DefaultAppDomain.powersh
        fr--r--r--                1 Mon Jan  1 00:00:00 1601    TDLN-2856-41
        fr--r--r--                1 Mon Jan  1 00:00:00 1601    TDLN-4196-41
        fr--r--r--                1 Mon Jan  1 00:00:00 1601    Winsock2\CatalogChangeListener-3f0-0
        IPC$                                                    READ ONLY       Remote IPC
```

SMB enumeration of all the shares and named pipes

Remediation Recommendations:

Disable null sessions on all SMB shares.

Informational

Domain User Enumeration

Threat Level: Informational

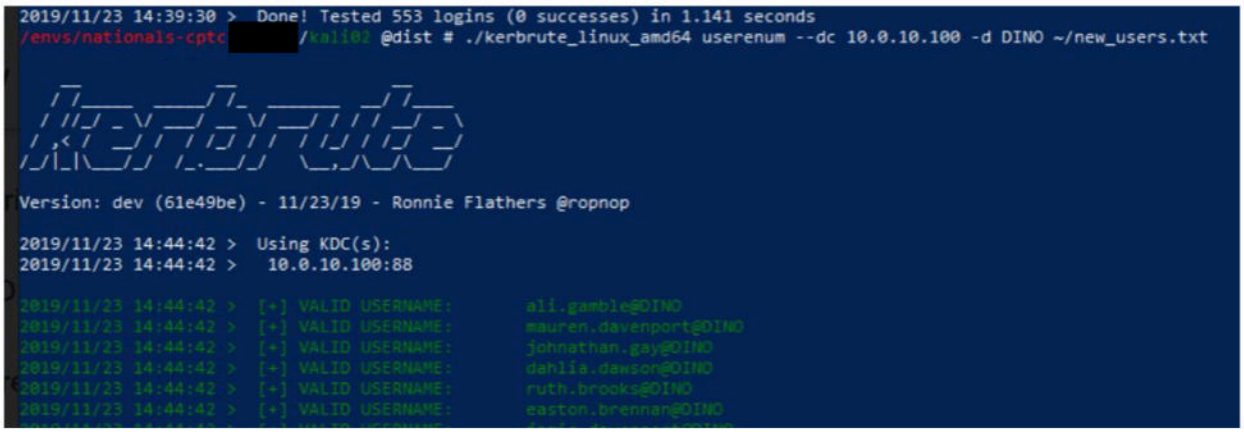
Description:
Our security engineers were able to use Kerberos to enumerate domain users. This was accomplished using a publicly available tool (Kerbrute) to differentiate responses to a preauthentication request between valid and invalid users. Note that this vulnerability was present in our previous engagement and has not been remediated since then.

Potential Business Impact:
This finding allows an attacker to gain usernames from a machine, increasing the likelihood of lateral movement within the network, a critical step for domain compromise.

Affected Host(s):
10.0.1.10:88 -- CORP-DC-01
10.0.10.100:88 -- GOTHAM-DC
10.0.11.100:88 -- METRO-DC
10.0.12.100:88 -- SPRING-DC

Exploitation Details:

An open PostGreSQL database on 10.0.2.100 contained extensive sensitive employee data, including 553 domain usernames. Our security engineers extracted these usernames from the database and used an open source tool (Kerbrute) to test them against the domain controllers in the dinobank.us domain. This resulted in the enumeration of 167 valid domain user accounts.



Enumeration of accounts on the dinobank.us domain using Kerbrute

Remediation Recommendations:

Firewall rules can be set up on the domain controllers that only allow trusted clients and servers to attempt pre-authentication. This prevents attackers on non-whitelisted hosts from enumerating domain user accounts.

Domain Admin Enumeration

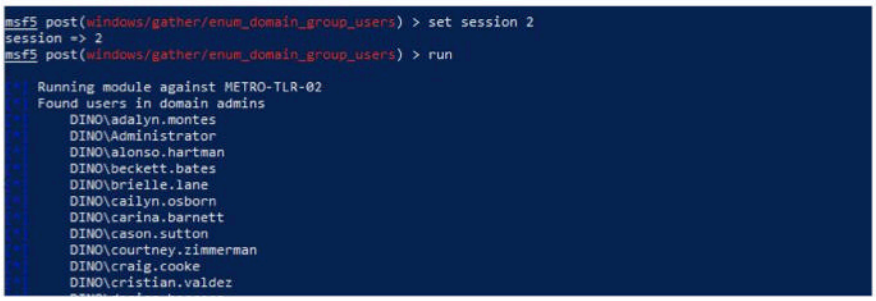
Threat Level: Informational

Description:
Our security engineers utilized a standard Metasploit module to enumerate domain administrators from a domain-joined host.

Potential Business Impact:
This finding allows a malicious individual to garner information on the domain administrators, and thus permits easier brute-forcing and targeted attacks.

Affected Host(s):
10.0.10.100 – GOTHAM-DC
10.0.11.100 – METRO-DC
10.0.12.100 – SPRING-DC

Exploitation Details:
After gaining local administrative credentials on Windows workstations, our security engineers were able to use PsExec to set up a meterpreter session on domain-joined Windows machines. They then ran a Metasploit module that enumerates the domain users in a particular domain group. By using this for the domain admin group, they were successful in enumerating 42 domain administrators.



Enumeration of domain administrator accounts using Metasploit

Remediation Recommendations:

Local administrator and system accounts should not be granted domain rights, such as those that allow querying the domain controller for information on domain groups. Implementing these restrictions would force attackers to possess domain credentials in order to be able to enumerate any information about the domain.

Conclusion

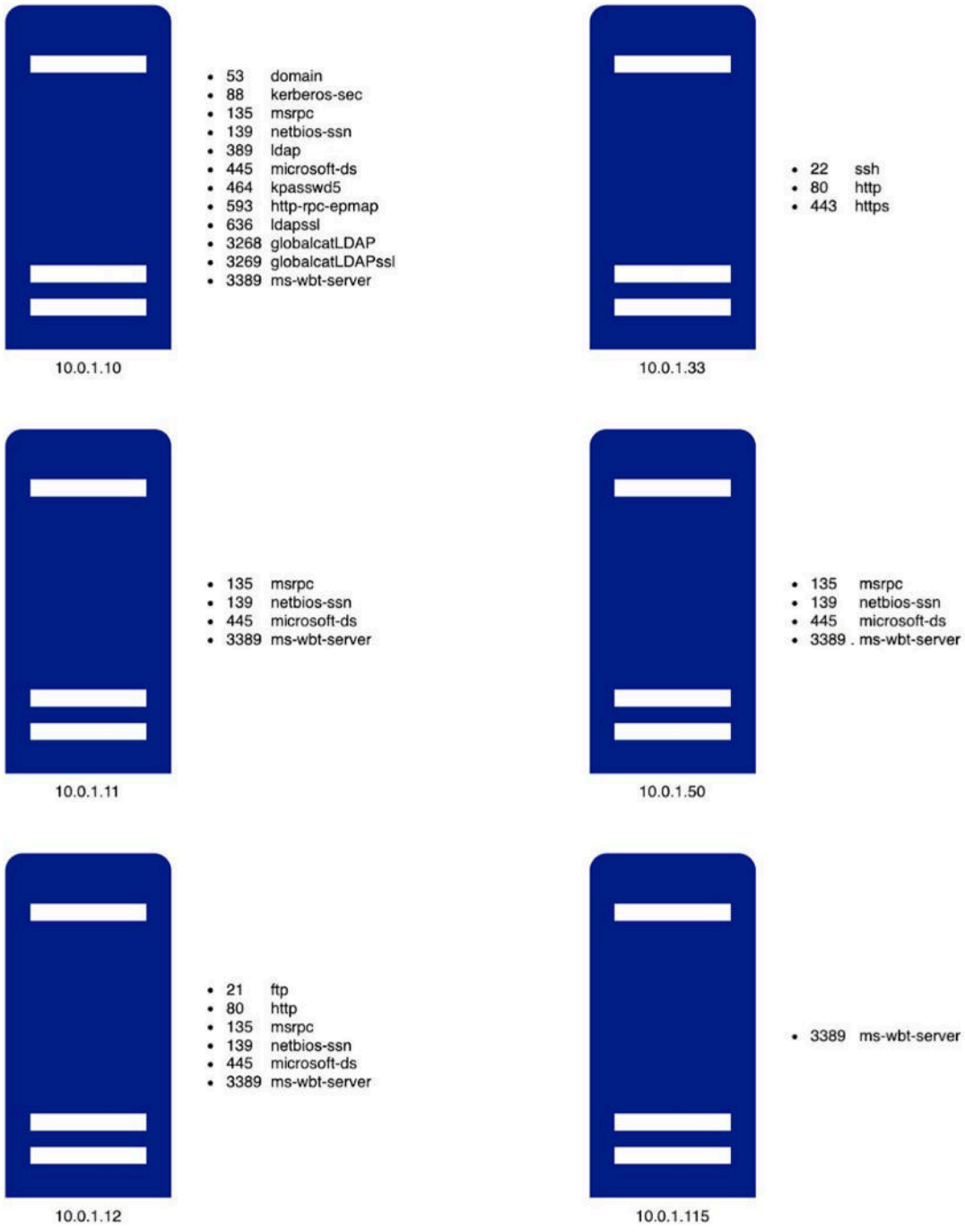
DinoBank is a unique, rapidly-growing fintech and cryptocurrency company, priding itself on its [pre]historical roots while also operating in a sector where security, public trust, and regulatory oversight are of paramount importance. While the company has taken steps to secure its network and privileged data, these require improvement given the presence of an MOU and regulatory attention. Our security consultants strongly suggest taking additional measures to increase the safety of the infrastructure and the reliability of employees, while reducing risk and mitigating harm, as described in the Executive Summary above. At this point in the company's growth, especially given the cryptocurrency sector's history of high-profile, trust-breaking data breaches and mass currency thefts, a single data breach could both legally imperil ongoing operations, and permanently tarnish its consumer-oriented reputation.

We encourage DinoBank to maintain constant vigilance and continue taking proactive responsibility for its network security and business risk mitigation. In the future, our security and compliance consultants are able to meet with the DinoBank team for additional follow-up engagements, where recommendations can be worked through in greater depth. DinoBank, as demonstrated by its contracting this assessment, clearly recognizes the value of a strong security posture, and though many of its present vulnerabilities and risks are pressing, they are still absolutely resolvable. Our security team is committed at every step of the way to continue working with the company to improve DinoBank's security. Together, we will transform DinoBank into a recognized leader in its field, while enabling it to set the pace and tone of tomorrow's cryptocurrency and fintech regulations.

Appendix

Network Diagrams

CIDR range 10.0.1.0/24 (CORP network):



10.0.1.20

- 25 smtp
- 80 http
- 81 hosts2-ns
- 135 msrpc
- 139 netbios-ssn
- 443 https
- 444 snpp
- 587 submission
- 593 http-rpc-epmap
- 808 ccproxy-http
- 1801 msmq
- 2103 zephyr-clt
- 2105 eklogin
- 2107 msmq-mgmt
- 2525 ms-v-worlds
- 3389 ms-wbt-server
- 3800 pwgpsi
- 3801 ibm-mgr
- 3828 neteh
- 5060 sip
- 6001 X11:1
- 6543 mythtv
- 6792 unknown

10.0.1.116

- 3389 ms-wbt-server

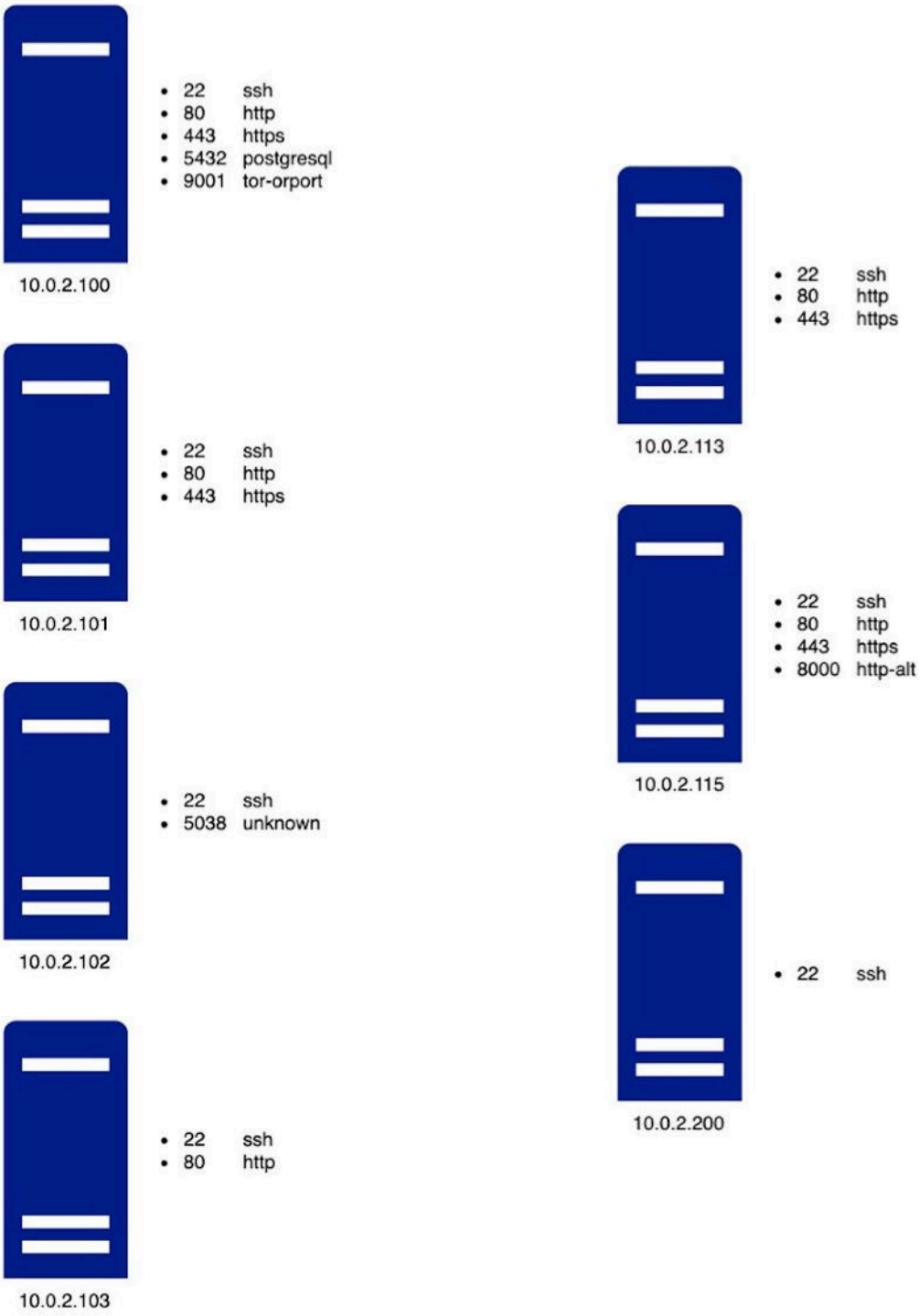
10.0.1.31

- 80 http
- 135 msrpc
- 139 netbios-ssn
- 445 microsoft-ds
- 3306 mysql
- 3389 ms-wbt-server

10.0.1.250

- 22 ssh
- 80 http
- 443 https

CIDR range 10.0.2.0/24 (BANK network):



CIDR range 10.0.10.0/24 (Gotham Branch):

- 53 domain
- 88 kerberos-sec
- 135 msrpc
- 139 netbios-ssn
- 389 ldap
- 445 microsoft-ds
- 464 kpasswd5
- 593 http-rpc-epmap
- 636 ldapssl
- 3268 globalcatLDAP
- 3269 globalcatLDAPssl
- 3389 ms-wbt-server

10.0.10.100

- 135 msrpc
- 139 netbios-ssn
- 445 microsoft-ds
- 3389 ms-wbt-server

10.0.10.203

- 135 msrpc
- 139 netbios-ssn
- 445 microsoft-ds
- 3389 ms-wbt-server

10.0.10.201

- 135 msrpc
- 139 netbios-ssn
- 445 microsoft-ds
- 3389 ms-wbt-server

10.0.10.208

- 135 msrpc
- 139 netbios-ssn
- 445 microsoft-ds
- 3389 ms-wbt-server

10.0.10.202

- 135 msrpc
- 139 netbios-ssn
- 445 microsoft-ds
- 3389 ms-wbt-server

10.0.10.209

CIDR range 10.0.11.0/24 (Metropolis Branch):

10.0.11.100

- 53 domain
- 88 kerberos-sec
- 135 msrpc
- 139 netbios-ssn
- 389 ldap
- 445 microsoft-ds
- 464 kpasswd5
- 593 http-rpc-epmap
- 636 ldapssl
- 3268 globalcatLDAP
- 3269 globalcatLDAPssl
- 3389 ms-wbt-server
- 5985 wsman
- 5986 wsmans
- 9389 adws
- 9971 unknown
- 47001 winrm
- 49664 unknown
- 49665 unknown
- 49673 unknown
- 49675 unknown
- 49678 unknown
- 49679 unknown
- 49683 unknown
- 49684 unknown
- 49689 unknown
- 49722 unknown
- 64144 unknown

10.0.11.202

- 135 msrpc
- 139 netbios-ssn
- 445 microsoft-ds
- 3389 ms-wbt-server
- 5985 wsman
- 5986 wsmans
- 9971 unknown
- 47001 winrm
- 49664 unknown
- 49665 unknown
- 49670 unknown
- 49672 unknown
- 49697 unknown
- 49707 unknown
- 49717 unknown
- 49730 unknown

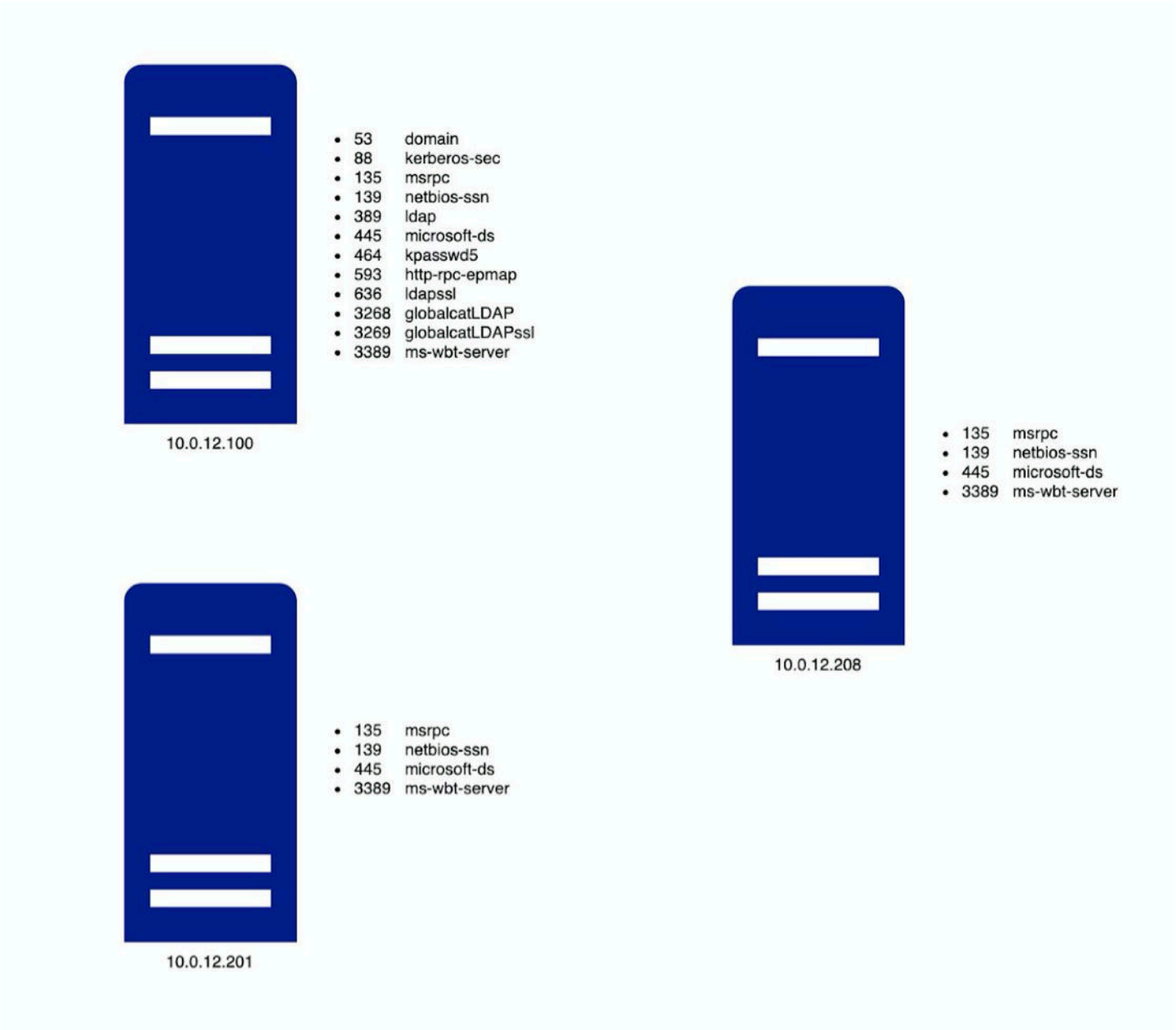
10.0.11.201

- 135 msrpc
- 139 netbios-ssn
- 445 microsoft-ds
- 3389 ms-wbt-server
- 5985 wsman
- 5986 wsmans
- 9971 unknown
- 47001 winrm
- 49664 unknown
- 49665 unknown
- 49666 unknown
- 49676 unknown
- 49714 unknown
- 49728 unknown
- 49735 unknown
- 49738 unknown

10.0.2.208

- 135 msrpc
- 139 netbios-ssn
- 445 microsoft-ds
- 3389 ms-wbt-server
- 5985 wsman
- 5986 wsmans
- 9971 unknown
- 47001 winrm
- 49664 unknown
- 49665 unknown
- 49670 unknown
- 49671 unknown
- 49693 unknown
- 49703 unknown
- 49723 unknown
- 49729 unknown

CIDR range 10.0.12.0/24 (Springfield Branch):



Tools

- Burp Suite Community Edition
- Amass
- CrackMapExec
- Dirbuster
- Enum4Linux
- Gobuster
- Hashcat
- Hydra
- Impacket
- JohnTheRipper
- Kerbrute
- Ldapsearch
- Metasploit
- Nmap
- Parameth
- Python PTY Shells
- SMBMap