**Team Structure and Roles**

1. **Team Lead (1 person)**: Coordinates the team's efforts, manages communication with the Gold Team and the incident response process.
2. **Network Security Specialists (2 people)**: Focuses on firewall management, monitoring network traffic, and handling intrusion detection systems (IDS)/intrusion prevention systems (IPS).
3. **System Administrators (2 people)**: Responsible for managing the operating systems, applying patches, and ensuring system integrity and availability. One focuses on Windows, and the other on Linux systems.
4. **Application Security Specialist (1 person)**: Manages web applications, ensures API security, and oversees the encryption of data transmissions.
5. **Incident Responder (1 person)**: Leads in incident detection, analysis, and response. Also, responsible for preparing and submitting incident reports.

**Strategy Overview**

**Pre-Competition Preparation**

* **Review Document**: Thoroughly review the provided packet to understand roles, network infrastructure, scoring, and rules.
* **Initial Setup Checklist**: Create a checklist based on the packet for initial actions such as password changes, multi-factor authentication setup, and network diagram review.
* **Tool Familiarization**: Ensure all team members are comfortable with the provided tools like puTTY, Remmina, Chrome, and KeePass.
* **Script Preparation**: Develop and review scripts for automating routine tasks, such as log analysis and alerting for suspicious activities. Ensure compliance with the competition's scripting rules.

**During Competition**

* **Initial Assessment and Configuration**:
  + Securely change initial passwords and report changes as required.
  + Verify multi-factor authentication setup for all critical accounts.
  + Review and understand the network diagram to identify key assets and potential vulnerabilities.
* **Continuous Monitoring**:
  + Set up network monitoring tools to alert on unusual traffic patterns or potential attacks.
  + Regularly review system and application logs for signs of unauthorized access or other security issues.
* **Vulnerability Management**:
  + Conduct an initial scan of all systems to identify and prioritize vulnerabilities.
  + Apply necessary patches and fixes to vulnerable systems, ensuring to minimize downtime.
* **Incident Handling**:
  + Establish a clear process for incident detection, analysis, response, and reporting.
  + Practice writing concise, coherent incident reports as they contribute to the team's score.
* **Communication**:
  + Maintain continuous communication within the team and with the Gold Team for reporting issues and resolving queries.
  + Monitor the Chief Medical Officer's email account as required, ensuring to identify and respond to injects promptly.

**Key Focus Areas**

* **Data Protection**: Prioritize the security of PII and PHI by ensuring data encryption both at rest and in transit.
* **Firewall and Access Controls**: Regularly update firewall rules to restrict unnecessary access and secure the perimeter.
* **Backups and Recovery**: Although direct VM backups are not allowed, ensure critical data within the VMs is backed up following best practices.
* **Compliance with Rules**: Avoid engaging in active defense or out-of-band actions that could lead to penalties.

**Continuous Improvement**

* **After-Action Reviews**: After each major incident or at the end of each day, conduct a brief review to discuss what went well, what didn't, and how processes can be improved.
* **Adapt and Overcome**: Stay flexible and ready to adapt the strategy based on the evolving scenario of the competition.

**1. Initial Assessment and Configuration**

* **Securely Change Initial Passwords**: Use complex passwords that comply with security best practices (a mix of uppercase, lowercase, numbers, and symbols). Utilize a password manager like KeePass for secure storage. Ensure all team members change their passwords at the start of the competition and report these changes to the Gold Team to avoid affecting the scoring system.
* **Verify Multi-Factor Authentication Setup**: Follow the competition guidelines to set up MFA for all critical accounts. This could involve using a personal email accessible via a browser. Ensure all team members understand this process and complete it before engaging in other activities.
* **Review Network Diagram**: Familiarize yourself with the provided network diagram to identify critical assets, potential choke points, and segmentation opportunities. This will aid in prioritizing security measures and understanding the flow of data within the network.

**2. Continuous Monitoring**

* **Network Monitoring Tools Setup**: Configure tools such as IDS/IPS for monitoring network traffic. Use puTTY for secure remote access and monitor logs for suspicious activity. Set up alerts for unusual outbound traffic, multiple failed login attempts, and other indicators of compromise.
* **Log Review**: Regularly check system and application logs for unauthorized access attempts, unexpected changes, or errors. Develop scripts to automate the collection and analysis of logs, highlighting potential security incidents for further investigation.

**3. Vulnerability Management**

* **Initial System Scanning**: Use vulnerability scanning tools compatible with both Windows and Linux systems to identify and prioritize vulnerabilities. Tools like OpenVAS or Nessus can be used for this purpose, focusing on high-risk vulnerabilities first.
* **Patch Management**: Apply patches to identified vulnerabilities, starting with the most critical ones. For Windows systems, use Windows Update or WSUS for batch updating; for Linux systems, use package managers like apt or yum. Test patches in a non-critical system when possible to ensure they don’t affect system stability.

**4. Incident Handling**

* **Incident Response Process**: Establish a clear incident response protocol including detection, containment, eradication, and recovery phases. Use a standardized incident report template that includes the incident’s description, affected systems, the source of the attack, remediation steps, and a recovery plan.
* **Incident Reporting**: Train all team members on how to write and submit incident reports. Ensure reports are detailed, coherent, and professional. Submit reports through the Gold Team Help Desk following the specified format.

**5. Communication**

* **Team Coordination**: Utilize the Stemrocks account for internal team communication, keeping track of tasks and incidents. Assign someone to monitor the Chief Medical Officer’s email for business tasks and other critical communications.
* **Gold Team Interaction**: Communicate with the Gold Team through the Help Desk for password changes, firewall requests, recovery requests, and incident reporting. Use clear, concise language and provide all required information to avoid delays.

**Continuous Improvement**

* **After-Action Reviews (AARs)**: Conduct brief AARs to discuss the team’s performance, what strategies were effective, and what could be improved. Focus on both technical actions and team coordination.
* **Flexibility and Adaptation**: Stay adaptable to new information or changes in the competition scenario. Be prepared to adjust your strategy based on feedback from AARs and new threats identified during the competition.

**Specific Tools and Techniques**

* **puTTY and Remmina**: Use for secure remote access. Configure puTTY with SSH keys for additional security when accessing Linux servers. Remmina can be set up for RDP sessions with strong encryption for Windows systems.
* **KeePass**: Store all passwords securely. Create a shared database for team access, but ensure it's stored securely and access is controlled.
* **IDS/IPS Configuration**: Use Snort or Suricata for intrusion detection. Configure rules to alert on known attack signatures and anomalous traffic patterns.
* **Log Analysis Scripts**: Write scripts in Python or PowerShell to automate log analysis. Focus on extracting and alerting on key indicators of compromise, such as unusual login times, failed authentication attempts, or large data transfers.

Securing the Domain Controller (DC) is crucial in any network, especially in competitive environments like cybersecurity competitions where the DC is a prime target. Group Policy Objects (GPOs) are an effective way to enforce security policies across your domain. Below is a list of essential GPOs to secure your Domain Controller:

**1. Password Policy**

* **Enforce password history**: 24 passwords remembered.
* **Maximum password age**: 60 days.
* **Minimum password age**: 1 day.
* **Minimum password length**: 14 characters.
* **Password must meet complexity requirements**: Enabled.
* **Store passwords using reversible encryption**: Disabled.

**2. Account Lockout Policy**

* **Account lockout duration**: 15 minutes.
* **Account lockout threshold**: 5 invalid login attempts.
* **Reset account lockout counter after**: 15 minutes.

**3. Audit Policy**

* **Audit account logon events**: Success, Failure.
* **Audit account management**: Success, Failure.
* **Audit directory service access**: Success, Failure (if applicable).
* **Audit logon events**: Success, Failure.
* **Audit object access**: Failure.
* **Audit policy change**: Success, Failure.
* **Audit privilege use**: Failure.
* **Audit system events**: Success, Failure.

**4. User Rights Assignment**

* **Deny access to this computer from the network**: to guest accounts.
* **Deny log on as a batch job**: to guest accounts.
* **Deny log on as a service**: to guest accounts.
* **Deny log on locally**: to guest accounts.
* **Deny log on through Remote Desktop Services**: to guest accounts and local accounts (only allow necessary administrative accounts).

**5. Security Options**

* **Accounts: Administrator account status**: Enabled.
* **Accounts: Guest account status**: Disabled.
* **Accounts: Limit local account use of blank passwords to console logon only**: Enabled.
* **Audit: Shut down system immediately if unable to log security audits**: Enabled.
* **DCOM: Machine Access Restrictions in Security Descriptor Definition Language (SDDL) syntax**: Properly configure to limit access.
* **DCOM: Machine Launch Restrictions in SDDL syntax**: Properly configure to limit launch permissions.

**6. Event Log Settings**

* **Maximum security log size**: 196608 KB or larger based on disk space.
* **Retention method for security log**: Overwrite events as needed.

**7. Kerberos Policy**

* **Enforce user logon restrictions**: Enabled.
* **Maximum lifetime for service ticket**: 600 minutes.
* **Maximum lifetime for user ticket**: 10 hours.
* **Maximum lifetime for user ticket renewal**: 7 days.
* **Maximum tolerance for computer clock synchronization**: 5 minutes.

**Implementation Tips:**

* **Regular Review and Update**: Security policies should be regularly reviewed and updated based on the evolving threat landscape and organizational changes.
* **Least Privilege**: Always apply the principle of least privilege, ensuring that accounts have only the permissions necessary to perform their roles.
* **Testing**: Test new GPOs in a controlled environment before deploying them broadly, to avoid unintended disruptions.
* **Documentation**: Keep detailed documentation of all GPOs implemented, including the rationale for each policy, to facilitate audits and troubleshooting.

Implementing these GPOs will significantly enhance the security posture of your Domain Controller, but remember, security is not a set-it-and-forget-it task. Continuous monitoring, updating, and education are key to maintaining a secure environment.

**Example Scripts**

**1. Script to Create and Configure a Password Policy GPO**

This script creates a new GPO for password policies and configures it with recommended settings.

powershell

# Import Active Directory and GroupPolicy modules

Import-Module ActiveDirectory

Import-Module GroupPolicy

# Define GPO Name

$GPOName = "SecurePasswordPolicy"

# Create the GPO

New-GPO -Name $GPOName -Comment "GPO to enforce secure password policies"

# Configure Password Policy settings

Set-GPRegistryValue -Name $GPOName -Key "HKLM\Software\Policies\Microsoft\Windows\CurrentVersion\Policies\System" -ValueName "MinimumPasswordLength" -Type DWord -Value 14

Set-GPRegistryValue -Name $GPOName -Key "HKLM\Software\Policies\Microsoft\Windows\CurrentVersion\Policies\System" -ValueName "PasswordComplexity" -Type DWord -Value 1

# Link the GPO to the Domain Controllers OU

$linkOU = "OU=Domain Controllers,DC=yourdomain,DC=com"

New-GPLink -Name $GPOName -Target $linkOU

**2. Script for Account Lockout Policy**

powershell

$GPOName = "AccountLockoutPolicy"

# Create the GPO

New-GPO -Name $GPOName -Comment "GPO for Account Lockout Policy"

# Configure Account Lockout Policy settings

Set-GPRegistryValue -Name $GPOName -Key "HKLM\Software\Policies\Microsoft\Windows\CurrentVersion\Policies\System" -ValueName "LockoutBadCount" -Type DWord -Value 5

Set-GPRegistryValue -Name $GPOName -Key "HKLM\Software\Policies\Microsoft\Windows\CurrentVersion\Policies\System" -ValueName "ResetLockoutCount" -Type DWord -Value 15

Set-GPRegistryValue -Name $GPOName -Key "HKLM\Software\Policies\Microsoft\Windows\CurrentVersion\Policies\System" -ValueName "LockoutDuration" -Type DWord -Value 15

# Link the GPO

New-GPLink -Name $GPOName -Target $linkOU

**Note**

* These scripts use Set-GPRegistryValue cmdlet to configure policy settings, which requires specifying the policy setting's registry path. This approach is suitable for some settings but may not work for all types of GPO configurations, especially those that are not directly mapped to registry settings (like Kerberos or Audit policies).
* Always replace "DC=yourdomain,DC=com" with your actual domain's DN (Distinguished Name).

**Advanced Configuration**

For more advanced GPO settings that can't be directly applied via registry keys (like Kerberos policies, advanced audit policies, etc.), you'll need to edit GPO templates or use ADMX-backed policies, which are beyond the scope of simple PowerShell scripting due to their complexity and potential impacts.

**Conclusion**

These example scripts are a starting point. Due to the wide range and complexity of GPO settings, not all settings can or should be scripted without a deep understanding of the potential impacts. Always consult with your IT security and Active Directory experts before making bulk changes via scripts.

For comprehensive GPO management, including settings that aren't directly accessible via simple registry edits or require complex configurations, administrators often rely on a combination of PowerShell for scriptable tasks and manual configuration through the Group Policy Management Console (GPMC) for more nuanced settings. This approach allows for the precision and review necessary for critical security settings.

**Addressing the Gaps**

For areas not covered by direct scripting like Kerberos policy, advanced audit policies, and some user rights assignments, you would typically:

* **Use the GPMC**: For complex policies, especially those involving security settings not tied to a specific registry key (e.g., advanced auditing, Kerberos policies), using the GPMC provides a UI that ensures you're configuring policies correctly. This approach also allows you to review settings before applying them.
* **Employ Security Templates**: Windows provides security templates that can be imported into GPOs. These templates (.inf files) offer a pre-set collection of security settings that can be applied to a GPO. You can modify these templates according to your requirements before importing them.
* **Leverage Advanced PowerShell and ADMX Templates**: For a more programmatic approach, PowerShell can be used in conjunction with ADMX-backed policies. This involves using the Set-GPRegistryValue cmdlet for registry-based policies and the LGPO.exe tool or similar for importing/exporting GPO settings that aren't registry-based.

**Example of Advanced Scripting**

For settings that do not directly map to registry keys, like enabling specific audit policies or configuring user rights assignments, the approach requires interacting with the policy definitions directly or using templates. Here’s how you might approach a more complex scenario with PowerShell:

**Importing a Security Template to a GPO**

powershell

# Define the GPO and template path

$GPOName = "CustomSecurityPolicy"

$templatePath = "C:\Path\To\SecurityTemplate.inf"

# Import the template to the GPO

Import-GPO -BackupId $GPOName -TargetName $GPOName -Path $templatePath

# Assuming the GPO is already linked; if not, use New-GPLink as demonstrated previously

This script assumes you've created a security template that aligns with your security policy requirements. The Import-GPO cmdlet is used illustratively here; you would typically use tools like LGPO.exe for importing settings not directly exposed through PowerShell cmdlets.

Creating a comprehensive security policy involves both scripting for automatable tasks and manual configurations for more complex settings. Below, I provide a structured guide that encompasses both approaches, ensuring a thorough implementation of the GPO suggestions provided earlier.

**Part 1: Automated Scripting with PowerShell**

**1. Prepare Your Environment**

* Ensure you have the necessary permissions to modify GPOs.
* Install the Group Policy Management feature and the Active Directory module for PowerShell if not already present.

**2. Script Execution**

* Test the provided PowerShell scripts in a non-production environment first. Modify them as needed for your domain specifics.
* For each script:
  + Open PowerShell with administrative privileges.
  + Execute the scripts provided earlier to set up basic GPO configurations like Password Policy and Account Lockout Policy.

**Part 2: Manual Configuration via GPMC**

**1. Accessing Group Policy Management Console (GPMC)**

* From the Server Manager dashboard, navigate to Tools > Group Policy Management.

**2. Creating and Managing GPOs**

**Audit Policy**

* Navigate to your Domain Controllers OU.
* Right-click the OU and select “Create a GPO in this domain, and Link it here…”
* Name the GPO (e.g., “Audit Policy GPO”).
* Right-click the newly created GPO and select “Edit…”
* Navigate to Computer Configuration > Policies > Windows Settings > Security Settings > Advanced Audit Policy Configuration > Audit Policies.
* Configure each audit policy as needed, ensuring both "Success" and "Failure" are selected for critical categories.

**User Rights Assignment & Security Options**

* Within the same GPO Editor window:
* Navigate to Computer Configuration > Policies > Windows Settings > Security Settings > Local Policies > User Rights Assignment.
* Configure settings such as “Deny access to this computer from the network” for guests and other non-essential accounts.
* For Security Options, navigate to Computer Configuration > Policies > Windows Settings > Security Settings > Local Policies > Security Options.
* Adjust settings such as “Accounts: Administrator account status” to Enabled and “Accounts: Guest account status” to Disabled.

**Kerberos Policy**

* Still in the GPO Editor:
* Navigate to Computer Configuration > Policies > Windows Settings > Security Settings > Account Policies > Kerberos Policy.
* Adjust the settings to match recommended values for ticket lifetimes and tolerances.

**3. Linking GPOs to the Appropriate OUs**

* Ensure each GPO is linked to the correct OU, such as your Domain Controllers OU for policies specifically designed for DC security.

**4. Enforcing and Testing GPOs**

* Use the “Group Policy Update” feature in the GPMC to push the policies immediately or wait for the natural replication cycle.
* Test the policy effects on a non-critical system before widespread deployment.

**Part 3: Additional Considerations**

**Document Your Policies**

* Keep detailed documentation of all GPO settings applied, including rationale and expected security outcomes.

**Regular Review and Update Cycle**

* Establish a regular review cycle for your GPO settings to adapt to new threats and organizational changes.

Prioritizing the steps for implementing Group Policy Objects (GPOs) to secure a domain environment efficiently and effectively involves balancing immediate security benefits against potential impacts on system functionality and user productivity. Here’s a structured approach, ordered by priority:

**1. Initial Setup and Environment Preparation**

* **Permissions Check**: Ensure you have administrative privileges necessary for creating and modifying GPOs.
* **Install Necessary Tools**: Verify that the Group Policy Management feature and the Active Directory module for PowerShell are installed.
* **Backup Existing GPOs**: Before making changes, back up all existing GPOs to avoid potential loss of settings.

**2. Critical Security Settings**

These settings directly impact the security posture of your environment and should be implemented first.

* **Password Policy**: Enforce strong password policies to protect against brute-force attacks.
* **Account Lockout Policy**: Prevent unauthorized access by configuring account lockout settings.
* **User Rights Assignment**: Limit system access and privileges to minimize the potential impact of compromised accounts.

**3. Audit and Monitoring**

Implementing comprehensive auditing and monitoring provides visibility into actions within your environment, crucial for identifying and responding to potential security incidents.

* **Audit Policy**: Configure audit policies to log successful and failed security events, such as account logon attempts and policy changes.

**4. Security Options**

These settings help further secure the operating environment and reduce the attack surface.

* **Security Options**: Configure security settings, such as disabling the guest account and securing the Administrator account.
* **Kerberos Policy**: Adjust Kerberos token settings to balance security and usability, ensuring that ticket lifetimes are not excessively long.

**5. Network and Communication Security**

Securing communication within your network prevents attackers from intercepting sensitive data.

* **Firewall Settings**: Though not directly a GPO setting, ensuring proper configuration of firewalls is crucial. Use PowerShell scripts or manual configurations to ensure firewalls are enabled and properly configured on Domain Controllers and other critical systems.

**6. Application Control Policies**

Minimizing the attack surface through application control policies can significantly enhance security.

* **AppLocker or Software Restriction Policies**: Implement application control policies to prevent the execution of unauthorized software.

**7. Additional Security Measures**

After the primary settings are in place, focus on additional measures that enhance security without significantly impacting system functionality.

* **Advanced Audit Policy Configuration**: Fine-tune audit policies for more detailed logging.
* **Encryption and Secure Communication Protocols**: Ensure that communication between domain members and controllers is secured using appropriate encryption protocols.

**8. Testing and Deployment**

* **Policy Testing**: Test new or modified GPOs in a non-production environment to ensure they do not negatively impact system functionality or user productivity.
* **Incremental Deployment**: Deploy GPOs incrementally, starting with a small, controlled group of systems or users, to monitor the impact and adjust as necessary.

**9. Documentation and Ongoing Management**

* **Documentation**: Maintain detailed records of all GPO settings applied, including the rationale and expected impact on security and operations.
* **Regular Review and Updates**: Establish a schedule for regularly reviewing and updating GPO settings to adapt to evolving threats, technologies, and organizational needs.

By following these prioritized steps, you can systematically strengthen your domain's security posture while maintaining a balance with operational requirements.

**AWS Firewall (Security Group) Configuration**

1. **Deny All Inbound Traffic by Default**
   * This is the default behavior of AWS Security Groups, so no explicit rule is needed for this.
2. **Allow Inbound Traffic from Blue Team Laptops**
   * **Rule Description**: Allow from Blue Team Laptops
   * **Protocol**: TCP/UDP (depending on your needs)
   * **Port Range**: Specify the port range based on the services you need to allow (e.g., 80, 443 for web traffic).
   * **Source IP**: <Blue\_Team\_Laptop\_IPs>/32 (Use the CIDR notation; replace <Blue\_Team\_Laptop\_IPs> with the actual IP addresses of your Blue Team laptops.)
3. **Allow Inbound Traffic from Unmanaged Switch**
   * Since an unmanaged switch does not have an IP address and operates at Layer 2, you cannot directly allow traffic from it based on an IP address. Instead, ensure the devices connected to the switch have their IP addresses explicitly allowed in the security group.
4. **Allow Inbound Traffic from Router/Firewall**
   * **Rule Description**: Allow from Router/Firewall
   * **Protocol**: TCP/UDP
   * **Port Range**: Specify the necessary ports.
   * **Source IP**: <Router\_Firewall\_IP>/32 (Replace <Router\_Firewall\_IP> with the IP address of your router/firewall.)

**Additional Considerations**

* **Egress Rules**: By default, AWS Security Groups allow all outbound traffic. Tailor the outbound rules to your specific requirements to ensure a least-privilege approach.
* **Specific Services**: If your environment requires specific services (SSH, RDP, HTTP/S, etc.), create individual rules for each service, specifying the correct protocol and port.
* **Dynamic IPs**: If the IPs of your devices might change (e.g., DHCP for laptops), consider using AWS Network Access Control Lists (NACLs) for broader subnet-level control or update the security group dynamically using scripts.
* **Update and Review**: Regularly review and update your firewall rules to adapt to changes in your network configuration and threat landscape.

**Implementing Rules in AWS**

To implement these rules in AWS:

1. **Navigate to the VPC Dashboard** in your AWS Management Console.
2. **Select Security Groups** and either create a new group or select an existing one to modify.
3. **Add Inbound Rules** using the "Edit inbound rules" option, applying the conceptual rules above with your specific details.

This approach ensures that your AWS-hosted resources are accessible only to specified devices, enhancing your security posture while supporting your operational needs.

To set up firewall rules that explicitly deny all traffic except from specified Blue Team devices in a scenario where you need to override potentially compromised settings (ignoring AWS defaults), you would create custom inbound and outbound rules to enforce strict access controls. Given that AWS Security Groups are stateful and default to allow all outbound traffic, and deny all inbound traffic not explicitly allowed, your focus will primarily be on defining precise inbound rules and revising outbound rules as necessary for your Blue Team scenario.

Here's a template for defining such rules within an AWS Security Group, taking into account the need for explicit denial except for trusted devices. When implementing, replace placeholder values with your actual IP addresses and service details.

**Steps to Reset and Configure AWS Security Group Rules**

**Step 1: Reset Inbound Rules**

1. **Deny All Inbound Traffic**:
   * Technically, AWS Security Groups don't support a "deny" action; they're "allow" lists. If a request doesn't match any rule, it's denied. So, you ensure only your explicit "allow" rules are present, effectively denying all other inbound traffic.

**Step 2: Allow Inbound Traffic from Specific Devices**

1. **Allow Traffic from Blue Team Laptops**:
   * **Description**: Blue Team Laptop Access
   * **Protocol**: Depending on your requirements (e.g., TCP for web traffic)
   * **Port Range**: Specific to your applications (e.g., 443 for HTTPS)
   * **Source**: <Laptop\_IP\_1>/32, <Laptop\_IP\_2>/32, etc.
2. **Allow Traffic from Specific Network Devices (e.g., a Managed Switch or Router)**:
   * **Description**: Network Device Access
   * **Protocol**: As required (e.g., TCP/UDP)
   * **Port Range**: Specific to the services needed
   * **Source**: <Device\_IP>/32

**Step 3: Review and Configure Outbound Rules**

While AWS Security Groups allow all outbound traffic by default, you may need to restrict this in a high-security scenario to ensure only specific communication is permitted from your AWS resources.

1. **Restrict Outbound Traffic to Necessary Destinations**:
   * **Description**: Outbound to Specific Destinations
   * **Protocol**: Specific to your needs
   * **Port Range**: As required
   * **Destination**: Define specific IPs or ranges for required external services or set to your internal network ranges to restrict outbound traffic to known destinations.

**Implementing in AWS Management Console**

1. **Open the VPC Dashboard**: Go to your AWS Management Console, navigate to VPC, or use the search bar to find VPC.
2. **Security Groups**: In the VPC Dashboard, find and select “Security Groups”.
3. **Edit or Create Security Group**: Choose to create a new security group or edit an existing group associated with your Blue Team resources.
4. **Configure Rules**:
   * For a new group, fill in the details and add your inbound and outbound rules according to the templates above.
   * For an existing group, you may need to remove existing rules before adding new ones to ensure you're starting from a known secure baseline.
5. **Apply Security Group**: Attach the security group to the relevant AWS resources (EC2 instances, RDS databases, etc.) that require protection.

**Additional Considerations**

* **Regularly Update**: As IP addresses for your devices may change (especially if they're not static), regularly review and update the security group rules to reflect any changes.
* **Logging and Monitoring**: Utilize AWS CloudTrail and VPC Flow Logs to monitor for unexpected requests or breaches, helping to adjust your rules as needed based on traffic patterns.

By following these steps, you create a focused security perimeter within your AWS environment that strictly limits access to predefined Blue Team devices, adding a layer of security against unauthorized access or compromised infrastructure.

Addressing security concerns across both Windows and Linux systems involves a multifaceted approach, focusing on identifying users, manipulating user accounts (e.g., adding, removing, or modifying them), and securing the systems from unauthorized access. Below, you'll find PowerShell scripts for Windows and Bash scripts for Linux that serve these purposes. These scripts are starting points and might need adjustments based on your specific environment and security requirements.

**Windows Systems via PowerShell**

**1. Identifying Users**

This PowerShell script lists all user accounts on the system.

powershell

Get-LocalUser | Select-Object Name, Enabled, LastLogon

**2. Manipulating User Accounts**

* **Adding a New User**

powershell

$UserName = "NewUser"

$Password = ConvertTo-SecureString "StrongPassword" -AsPlainText -Force

New-LocalUser -Name $UserName -Password $Password -FullName "New User" -Description "This is a new user."

* **Removing a User**

powershell

Remove-LocalUser -Name "OldUser"

* **Disabling a User Account**

powershell

Disable-LocalUser -Name "UserToDisable"

**3. Securing Systems from Unauthorized Access**

* **Enforcing Password Policy**

powershell

Set-LocalUser -Name "User" -PasswordNeverExpires $false -UserMayNotChangePassword $false -AccountExpires (New-TimeSpan -Days 90)

* **Locking Down RDP Access**

powershell

Set-ItemProperty -Path 'HKLM:\System\CurrentControlSet\Control\Terminal Server' -name "f

**Linux Systems via Bash**

**1. Identifying Users**

This Bash script lists all user accounts on the system along with their login shell.

bash

cut -d: -f1,7 /etc/passwd

**2. Manipulating User Accounts**

* **Adding a New User**

bash

sudo useradd -m -s /bin/bash NewUser

echo NewUser:StrongPassword | sudo chpasswd

* **Removing a User**

bash

sudo userdel -r OldUser

* **Disabling a User Account**

bash

sudo usermod -L UserToDisable

**3. Securing Systems from Unauthorized Access**

* **Enforcing Password Policy**

Modify the /etc/login.defs file for global password aging policies, and use chage for specific user accounts.

bash

sudo sed -i 's/^PASS\_MAX\_DAYS.\*/PASS\_MAX\_DAYS 90/' /etc/login.defs

sudo chage -M 90 UserName

* **Restricting SSH Access**

Edit the /etc/ssh/sshd\_config file to disable root login and restrict which users can log in.

bash

echo "PermitRootLogin no" | sudo tee -a /etc/ssh/sshd\_config

echo "AllowUsers NewUser" | sudo tee -a /etc/ssh/sshd\_config

sudo systemctl restart sshd

Detecting unauthorized access, backdoors, or reverse shells involves scrutinizing active network connections, running processes, and unusual system behavior. These scripts can help you start an investigation, but remember, sophisticated attackers might use techniques that evade simple detection methods. Continuous monitoring, anomaly detection, and employing a comprehensive security solution are critical for robust security.

**Windows Systems via PowerShell**

**Check for Unusual Network Connections**

This script lists established network connections and filters for unusual remote addresses or ports. You should review the list for anything out of the ordinary.

powershell

Get-NetTCPConnection | Where-Object { $\_.State -eq "Established" } | Select-Object -Property LocalAddress, LocalPort, RemoteAddress, RemotePort, State, OwningProcess | Format-Table -AutoSize

**List Running Processes**

Listing running processes can help identify unauthorized or suspicious processes.

powershell

Get-Process | Where-Object { $\_.Path -ne $null } | Select-Object Id, Name, Path | Format-Table -AutoSize

**Linux Systems via Bash**

**Check for Unusual Network Connections**

Using netstat or ss to list active connections. Look for unfamiliar external connections.

bash

sudo ss -tuln | awk 'NR>1 {print $1, $4, $5, $6}'

Or:

bash

sudo netstat -tulnp | grep -v '127.0.0.1\|::1\|0.0.0.0\|:::' | awk '{print $1, $4, $7}'

**List Running Processes**

This command lists all running processes, helping identify unknown or suspicious entries.

bash

ps aux | less

**Additional Detection Techniques**

* **Audit Logs**: Check system and application logs for unauthorized access attempts, failed login attempts, or other suspicious activities. For Windows, use Event Viewer; for Linux, check logs in /var/log/, such as auth.log or syslog.
* **File Integrity Monitoring**: Use tools like AIDE for Linux or the built-in Windows Defender on Windows to detect changes in critical system files and directories.
* **Check Scheduled Tasks and Cron Jobs**: Attackers often persist by adding tasks or jobs.
  + Windows: Get-ScheduledTask | where {$\_.State -eq "Ready"}
  + Linux: Check crontab entries with crontab -l and also examine /etc/cron.\* directories.
* **Review Installed Programs and Services**: Compare the current list of installed programs and services with a known baseline. Unexpected entries might indicate compromise.
  + Windows: Get-WmiObject -Class Win32\_Product | Select-Object -Property Name
  + Linux: Use package managers to list installed packages, e.g., dpkg -l on Debian-based systems.

**Post-Detection Actions**

If you detect unauthorized access or other signs of compromise:

* **Containment**: Disconnect affected systems from the network to prevent further spread.
* **Eradication**: Remove backdoors, reverse shells, and any malware found. This may involve deleting files, killing processes, and removing persistence mechanisms.
* **Recovery**: Restore affected systems from backups if necessary. Rebuild systems when backups are not trustworthy or available.
* **Post-Incident Analysis**: Conduct a thorough investigation to understand the breach's extent, the attacker's entry point, and the data or systems compromised.
* **Enhance Security Posture**: Update your security measures based on findings to prevent future breaches. This includes patching vulnerabilities, changing passwords, and improving monitoring and detection capabilities.

Anomaly detection is a crucial aspect of cybersecurity, helping identify unusual patterns that may indicate malicious activity. Various strategies can be applied, many of which can be automated or supported by scripts. Here's an overview of some effective strategies and how they can be implemented:

**1. Log Analysis**

* **Strategy**: Analyze logs from various sources (system logs, application logs, network logs) for unusual patterns, such as failed login attempts, unexpected access patterns, or large data transfers.
* **Scripting**: Scripts can automate the collection and preliminary analysis of logs, flagging unusual activity for further investigation. Tools like PowerShell on Windows and Bash on Linux can parse log files, while more sophisticated log analysis can leverage Python scripts combined with tools like Elasticsearch and Logstash.

**2. Network Traffic Monitoring**

* **Strategy**: Monitor network traffic for unusual volume, unexpected external connections, or the use of uncommon ports and protocols.
* **Scripting**: While low-level network monitoring typically requires specialized tools (like Wireshark, tcpdump, or commercial solutions), scripts can automate the alerting based on outputs from these tools or integrate with network monitoring APIs to fetch and analyze data.

**3. Baseline Behavior Modeling**

* **Strategy**: Establish a baseline of normal behavior for system metrics, user activities, and network traffic. Anomalies are detected as deviations from this baseline.
* **Scripting**: Scripts can collect data over time to establish baselines and use statistical methods to detect deviations. This could involve complex data analysis and might benefit from using Python with libraries like Pandas and SciPy for analysis.

**4. Machine Learning**

* **Strategy**: Use machine learning models to learn normal behavior and detect anomalies. This approach can adapt over time to detect subtle and complex anomalies.
* **Scripting**: Implementing machine learning models for anomaly detection typically requires advanced programming and data science skills. Python, with libraries like scikit-learn, TensorFlow, or PyTorch, is often used. Scripts can automate data collection and preprocessing, model training, and the deployment of models for ongoing monitoring.

**5. File Integrity Monitoring**

* **Strategy**: Monitor critical system and configuration files for unauthorized changes, which can indicate a compromise.
* **Scripting**: Tools like AIDE (Advanced Intrusion Detection Environment) for Linux or the Windows Security Log can track file changes. Scripts can interface with these tools to automate monitoring and alerts.

**6. Endpoint Behavior Analysis**

* **Strategy**: Monitor endpoints for unusual processes, high resource usage, or unauthorized configuration changes.
* **Scripting**: Scripts running on endpoints can collect information about processes, resource usage, and configurations, comparing them against known good baselines to detect anomalies.

**Example Script for Log Analysis (Bash)**

bash

#!/bin/bash

# Simple script to check for failed SSH login attempts

LOG\_FILE="/var/log/auth.log"

ALERT\_COUNT=100

FAILED\_ATTEMPTS=$(grep "Failed password" $LOG\_FILE | wc -l)

if [ $FAILED\_ATTEMPTS -gt $ALERT\_COUNT ]; then

echo "Alert: High number of failed login attempts detected!"

fi

**Security Considerations**

* **Privacy and Compliance**: Ensure that your anomaly detection methods comply with privacy laws and organizational policies.
* **Alert Fatigue**: Carefully tune alert thresholds to avoid overwhelming teams with false positives.
* **Continuous Improvement**: Regularly review and update your detection strategies and scripts to adapt to evolving threats.

Creating comprehensive scripts for each anomaly detection strategy involves a blend of system commands, data analysis, and possibly leveraging external tools or libraries for more sophisticated approaches. Below are basic scripts or script outlines for each strategy mentioned. These scripts serve as starting points; depending on the complexity and specific requirements of your environment, further customization and expansion will be necessary.

**1. Log Analysis**

**Simple Bash Script for Detecting Unusual SSH Login Attempts on Linux**

bash

#!/bin/bash

LOG\_PATH="/var/log/auth.log"

ALERT\_THRESHOLD=50

# Count the number of failed login attempts

FAILED\_LOGIN\_ATTEMPTS=$(grep "Failed password" $LOG\_PATH | wc -l)

if [ "$FAILED\_LOGIN\_ATTEMPTS" -gt "$ALERT\_THRESHOLD" ]; then

echo "Alert: High number of SSH failed login attempts detected: $FAILED\_LOGIN\_ATTEMPTS"

fi

**2. Network Traffic Monitoring**

Due to the complexity and the need for real-time data, network traffic monitoring often relies on tools like tcpdump, Wireshark, or nftables/iptables. Here's a concept for using tcpdump to capture and analyze traffic:

**Bash Concept for Monitoring Uncommon Ports with tcpdump**

bash

#!/bin/bash

UNCOMMON\_PORT="6667"

DURATION="60" # seconds

# Run tcpdump in the background for the specified duration and capture packets on the uncommon port

tcpdump port $UNCOMMON\_PORT -w /tmp/uncommon\_port\_traffic.pcap &

TCPDUMP\_PID=$!

sleep $DURATION

kill $TCPDUMP\_PID

# Analyze the capture: for demonstration, just count packets

PACKET\_COUNT=$(tcpdump -r /tmp/uncommon\_port\_traffic.pcap 2>/dev/null | wc -l)

echo "Packets captured on port $UNCOMMON\_PORT: $PACKET\_COUNT"

**3. Baseline Behavior Modeling**

Creating a baseline and detecting deviations could be complex, especially for scripting from scratch. Here's a Python concept using pandas for simplicity:

**Python Script for Basic Anomaly Detection Based on Historical Data**

python

import pandas as pd

# Load historical data to establish baseline

historical\_data = pd.read\_csv('historical\_data.csv')

# Assuming historical\_data has a 'login\_attempts' column

mean\_attempts = historical\_data['login\_attempts'].mean()

std\_attempts = historical\_data['login\_attempts'].std()

# Threshold for anomaly detection

threshold = mean\_attempts + 2 \* std\_attempts

# Current data

current\_data = pd.read\_csv('current\_data.csv')

current\_attempts = current\_data['login\_attempts'].iloc[-1]

if current\_attempts > threshold:

print("Anomaly Detected: Login attempts exceed expected range.")

**4. Machine Learning for Anomaly Detection**

Implementing a machine learning model for anomaly detection typically involves substantial data preparation, training, and validation steps. Here's a high-level Python concept using scikit-learn:

python

from sklearn.ensemble import IsolationForest

import pandas as pd

# Load your dataset

data = pd.read\_csv('network\_traffic\_data.csv')

# Assuming 'data' is a DataFrame with numerical features

model = IsolationForest(n\_estimators=100, contamination='auto')

model.fit(data)

# Predict anomalies (-1 labels anomalies)

predictions = model.predict(data)

data['anomaly'] = predictions

# Filter and report anomalies

anomalies = data[data['anomaly'] == -1]

print(f"Detected anomalies:\n{anomalies}")

**5. File Integrity Monitoring**

File integrity monitoring on Linux can be achieved with tools like AIDE or through simple hash comparison scripts for critical files:

**Bash Script for Simple File Integrity Check**

bash

#!/bin/bash

FILE\_PATH="/etc/important\_config.conf"

KNOWN\_GOOD\_HASH="known\_good\_hash\_here"

current\_hash=$(sha256sum $FILE\_PATH | awk '{print $1}')

if [ "$current\_hash" != "$KNOWN\_GOOD\_HASH" ]; then

echo "File integrity check failed for $FILE\_PATH"

fi

**6. Endpoint Behavior Analysis**

Monitoring endpoints for unusual processes or high resource usage can be started with scripts but often requires deeper system integration or third-party tools for comprehensive coverage.

**PowerShell Script for Listing Processes Exceeding CPU Usage Threshold**

powershell

$cpuThreshold = 80

Get-Process | Where-Object {$\_.CPU -gt $cpuThreshold} | Format-Table Name, CPU -AutoSize

**Security and Compliance**

When implementing these scripts:

* Always test in a controlled environment first.
* Adjust thresholds and parameters to suit your specific operational environment.
* Ensure compliance with legal and organizational policies, especially when processing logs or monitoring network traffic.

**Segmentation Strategy Based on the PCDC2024 Diagram:**

1. **Blue Team Network**:
   * This is likely where sensitive operations occur. It contains workstations and servers including Domain Controller, Web Server, and Mail Server.
   * **Segmentation**: Utilize internal firewalls or VLANs to segment the Blue Team Network further. For example, separate VLANs for servers and workstations.
   * **Access Controls**: Ensure strict ACLs on the router/firewall to control traffic between these VLANs.
2. **Gold Network**:
   * Since it's marked as "Red Team Out of Scope," this may contain critical management infrastructure.
   * **Segmentation**: Keep this network isolated from the Blue Team Network. Use a firewall to regulate any communication between the two, if necessary.
3. **Support Network**:
   * Contains the Scoring Engine and API Server, which should be isolated to prevent tampering or unauthorized access.
   * **Segmentation**: Consider placing the Support Network in a separate VLAN with ACLs restricting access to only what's necessary for scoring and API interaction.
4. **On-Prem Network**:
   * It's also marked as "Red Team Out of Scope" and includes infrastructure components like the Router/Firewall and the Unmanaged Switch.
   * **Segmentation**: While the unmanaged switch doesn't allow for VLANs, ensure the Router/Firewall has rules to segregate traffic from the On-Prem Network from the rest of the environment.
5. **AWS Firewall**:
   * This serves as the barrier between your internal networks and the internet.
   * **Segmentation**: Use the AWS firewall to create boundaries between the internet and your internal networks. Ensure it's configured to block all unnecessary inbound and outbound traffic, allowing only specific, required ports and protocols.
6. **General Rules for Implementation**:
   * Define **Inbound and Outbound Rules** carefully for each segment. Inbound rules should be strict, allowing only necessary traffic and blocking the rest. Outbound rules should ensure that compromised systems cannot communicate out to C2 servers or exfiltrate data.
   * **Monitoring and Logging**: Set up monitoring on each segment's entry and exit points. Collect and analyze logs to quickly detect unusual activities.
   * **Physical Security**: Ensure that the physical access to network devices is secured and controlled to prevent tampering.

**Technical Details:**

* **VLANs**: For example, assign VLAN 10 to the Blue Team Workstations and VLAN 20 to the Servers. Configure inter-VLAN routing with ACLs to control traffic between them.

perl

* # Example configuration on a Layer 3 switch
* vlan 10
* name Blue\_Team\_Workstations
* vlan 20
* name Blue\_Team\_Servers
* int vlan 10
* ip address 192.168.10.1 255.255.255.0
* int vlan 20
* ip address 192.168.20.1 255.255.255.0
* ip access-list Blue\_Team\_ACL
* permit ip 192.168.10.0 0.0.0.255 192.168.20.0 0.0.0.255
* **AWS Firewall Rules**: Leverage AWS Network ACLs for stateless filtering along with Security Groups for stateful filtering. Block all inbound traffic except for allowed IPs and ports for services. Outbound rules should limit traffic to known destinations.

Implementing these segmentation strategies will help minimize the risk of lateral movement within your network and ensure that critical assets and services are protected from potential threats, both internal and external. Remember, the effectiveness of network segmentation is highly dependent on proper configuration and regular review of rules and policies.

Here are some specific tools that blue teams often use to enhance their cybersecurity posture:

1. **SIEM Systems**: These are crucial for real-time data analysis and network activity monitoring. Some popular SIEM systems include:
   * **AlienVault**
   * **FortiSIEM** (formerly AccelOps)
   * **LogRhythm**
   * [**Microsoft Sentinel**](https://www.bing.com/aclick?ld=e8H4akD4fmZ3jFpw4bRbxpBjVUCUxoPVBvVpos5GMfvsX7nIFQuPcF-8RGNi_u78xOcKcss6cWbhsJUaBUslv4i1TQFsjbeVYMiXrQTxDEmZXaMwhhGYm7HNvDbin2P5fXXsNsNknEQUmxEhlrTBjvA1qSSqHSbl6LCsJJLHN1yNDEWFvQ&u=&rlid=3a0baacb10e91fd69091ccd3353610ab)[1](https://en.wikipedia.org/wiki/Blue_team_%28computer_security%29)
2. **Intrusion Detection/Prevention Systems (IDS/IPS)**: These tools help detect and prevent cyber threats. Examples include:
   * **Snort** (Open Source IDS)
   * **Suricata** (Open Source IDS/IPS)
3. **Endpoint Protection Solutions**: To protect endpoints, you might consider:
   * **McAfee Endpoint Security**
   * **Symantec Endpoint Protection**
4. **Vulnerability Scanners**: These tools can identify security weaknesses in your systems. Some options are:
   * **OpenVAS** (Open Source Vulnerability Scanner)
   * **Nessus Essentials**
5. **Network Traffic Analyzers**: To analyze network traffic, tools like **Wireshark** can be very useful.
6. **Threat Intelligence Platforms**: These platforms help you understand the threat landscape better. Examples include:
   * **Maltego**
   * **MISP** (Malware Information Sharing Platform)
   * **ThreatConnect**
7. **Incident Response Tools**: For incident response, consider:
   * **TheHive** (Open Source Incident Response Platform)
   * **GRR Rapid Response**
8. **Forensic Tools**: For digital forensics, tools like **Autopsy** or **Sleuth Kit** can be invaluable.
9. **Phishing Defense**: Tools like **PhishTank** or **Cofense PhishMe** can help train employees to recognize phishing attempts.
10. **Data Visualization and Exploration**: For visualizing data, you might use:
    * **Kibana**
    * **Grafana**