Securing Ubuntu Linux in the IU Environment

The supported Linux distribution in the School of Informatics, Computing, and Engineering is Red Hat Enterprise Linux per Supported Linux Configurations. However, the school recognizes that certain projects will have dependencies on other Linux distributions and we make allowances for this per the IT Policy: Administrator Access and Self-Managed Systems. This page is intended to provide guidance to administrators of Ubuntu-based self-manged systems so they remain in compliance with IU IT policies aimed at keeping system secure. In some cases, instructions may be specific to a particular version of Ubuntu so you may have to make changes based on the version you are running.

1. **Identify Data Types** - It is critical that you evaluate the types of data you will be storing, transmitting, or manipulating on your self-managed system. If this involves any critical or restricted sensitive data, you MUST get prior approval before proceeding. Please see Sensitive Data Policies and Email Encryption and let us know before proceeding if you will need to store, transmit, or manipulate any sensitive data.

2. **OS Version Selection** - There are various versions of Ubuntu available and we strongly recommend selecting a Long-Term Support (**LTS**) version of Ubuntu. This way your version of the OS will get security patches for as long a possible without having to make a major OS version upgrade. You can run the **ubuntu-support-status** command to see the status of the system and end of support dates.

3. **Mobile Device Whole Disk Encryption** - If you are installing Ubuntu on a laptop or other mobile device, you must use whole disk encryption to be in compliance with the Mobile Device Security Standard, Policy IT-12.1. This is simple to do during the initial OS installation by just selecting the option. For additional information see the Linux section of Mobile Device Security Standards.

4. **Manual Security Updates** - You will want to set up automatic installation of security updates (per the next item below) but if you want to manually update your system you can see what updates are needed and update the system with:

   ```
   sudo /usr/lib/update-notifier/apt-check --human-readable
   sudo apt-get update
   sudo apt-get upgrade
   sudo apt-get dist-upgrade
   ```

   See the apt-get man page for details on the various commands and exactly what they do.

5. **Automatic Security Updates**- During the installation of Ubuntu, you will be asked if you want automated updates. You must select the option to apply security updates automatically. If you did not select this option during the initial installation, please enable it per the Ubuntu Automatic Updates Documentation. You can run "**sudo debconf-show unattended-upgrades**" to see if the automatic updates are enabled and you can reconfigure it with "**sudo dpkg-reconfigure -plow unattended-upgrades**".

   Once this is set up, you should see something like the following from debconf-show:

   ```
   $ sudo debconf-show unattended-upgrades
   * unattended-upgrades/origins_pattern: "origin=Debian,codename=${distro_codename},label=Debian-Security";
   * unattended-upgrades/enable_auto_updates: true
   ```

6. **Automatic Removal of Old Kernels** - You will also want to configure things so that unused packages are automatically uninstalled. If you don't do this then it is just a matter of time before your /boot partition fills up and which can cause various other problems. One problem it will cause is that updates will then fail and your system will no longer get security updates automatically. To enable the auto-removal of old kernel packages, edit the file **/etc/apt/apt.conf.d/50unattended-upgrades** and change these lines:

   ```
   //Unattended-Upgrade::Remove-Unused-Kernel-Packages "false";

   //Unattended-Upgrade::Remove-Unused-Dependencies "false";
   ```

   to look like this (uncomment the lines and set to true)

   ```
   Unattended-Upgrade::Remove-Unused-Kernel-Packages "true";

   Unattended-Upgrade::Remove-Unused-Dependencies "true";
   ```

   Note that the first line in this example may only be there for Ubuntu 18.04
   Reference: Ubuntu Community: Remove Old Kernels

7. **Account Passwords** - To be in compliance with IU policy, all account passwords must comply with the IU Passphrase Guidelines, including the 15 character minimum length. This includes the initial account you set up at install time and any other accounts you might add after the installation. In addition, account passwords must be changed no less frequently than every 2 years to be in compliance with IU policy. One good way to say in compliance with password guidelines is to set the system up so it uses the IU passphrase for account authentication. This is simple to do by just installing these packages:

   ```
   sudo apt-get install heimdal-clients libpam-heimdal
   ```

When promoted for the kerberos domain, enter "ADS.IU.EDU".

8. **Admin Access** - Normal day-to-day usage of the system must be done using non-privileged (ie. non-root) accounts. When elevated privileges are needed, sudo will be use. This is the default mode of operation in Ubuntu so should not be a problem. However, you are discouraged from routinely doing something like "**sudo bash**" to get a root shell when you can just run individual commands via sudo.

9. **Encryption Requirements** - Any service that requires logins over the network must be encrypted. So, for example, you must use ssh and sftp and not something like ftp that sends login information in cleartext. This also includes web applications that require login access so such sites must use https/SSL.

10. **Firewall Implementation** - One of the most important security mechanisms is the implementation of a proper firewall. In the linux world, that will likely be either iptables or firewalld. With Ubuntu, UFW (Uncomplicated Firewall) is a popular frontend for managing iptables firewall rules. There is a good guide to using UFW here:

    Uncomplicated Firewall Ubuntu Wiki

    An alternate method is to use iptables directly. Here are some cookbook example of setting up iptables with Ubuntu 18.04 or 16.04 LTS:

    Step 1: Set up the iptables rules - Here are 3 cookbook examples for 3 common use-cases

    Example 1: No Ports Open

    ```
    sudo iptables -A INPUT -m conntrack --ctstate ESTABLISHED,RELATED -j ACCEPT
    sudo iptables -I INPUT 1 -i lo -j ACCEPT
    sudo iptables -A INPUT -p icmp --icmp-type echo-request -j ACCEPT
    sudo iptables -P INPUT DROP

    ## See Step 2 below to make these changes permanent
    ```

    Example 2: Port 22/ssh Open To IU Only

    ```
    sudo iptables -A INPUT -m conntrack --ctstate ESTABLISHED,RELATED -j ACCEPT
    sudo iptables -A INPUT -p tcp -s 129.79.0.0/16 --dport 22 -j ACCEPT
    sudo iptables -A INPUT -p tcp -s 10.79.0.0/16 --dport 22 -j ACCEPT
    sudo iptables -A INPUT -p tcp -s 156.56.0.0/16 --dport 22 -j ACCEPT
    sudo iptables -A INPUT -p tcp -s 10.56.0.0/16 --dport 22 -j ACCEPT
    sudo iptables -A INPUT -p tcp -s 140.182.0.0/16 --dport 22 -j ACCEPT
    sudo iptables -A INPUT -p tcp -s 149.159.0.0/16 --dport 22 -j ACCEPT
    sudo iptables -A INPUT -p tcp -s 149.160.0.0/16 --dport 22 -j ACCEPT
    sudo iptables -A INPUT -p tcp -s 149.161.0.0/16 --dport 22 -j ACCEPT
    sudo iptables -A INPUT -p tcp -s 149.165.0.0/16 --dport 22 -j ACCEPT
    sudo iptables -I INPUT 1 -i lo -j ACCEPT
    sudo iptables -A INPUT -p icmp --icmp-type echo-request -j ACCEPT
    sudo iptables -P INPUT DROP

    ## See Step 2 below to make these changes permanent

    NOTE: This opens up ssh access to the primary IUB subnets, including IU Secure wireless and the IU VPN.
    If you need to ssh to the system from outside of the IUB network then you just need to connect to the IU
    VPN first.  See also the sshd section in the 'Securing Services' item below if you are going to expose
    sshd on the network.
    ```

    If you configure ssh to be open to the world as noted in the following section you MUST take additional steps to secure ssh as noted in Block Brute-Force SSH Attacks

Example 3: Port 22/ssh Open To The World (Not Recommended, See Next Section)

```
We strongly discourage you from opening port 22 to the world.  That opens the system up to all types of
attacks and settings things up this way requires that you take steps to mitigate the risks as noted in
the following section.

sudo iptables -A INPUT -m conntrack --ctstate ESTABLISHED,RELATED -j ACCEPT
sudo iptables -A INPUT -p tcp --dport 22 -j ACCEPT
sudo iptables -I INPUT 1 -i lo -j ACCEPT
sudo iptables -A INPUT -p icmp --icmp-type echo-request -j ACCEPT
sudo iptables -P INPUT DROP

## See Step 2 below to make these changes permanent

NOTE:  If you open port 22 to the world (which is strongly discouraged) then you must take precautions
as noted in the next section.  If you leave it only open to IU then you can use the IU VPN to connect
when outside of IU which is preferable (as in the previous example).  See also the sshd section in the
'Securing Services' item below if you are going to expose sshd on the network.
```

Step 2: Install iptables-persistent to manage and save the config

You can use iptables-persistent to save and manage the iptables config. Just install that and accept the defaults to save out the current configuration:

```
sudo apt-get update
sudo apt-get install iptables-persistent

OR, if this package was already installed run:

sudo netfilter-persistent save

NOTE: Rules will be saved to /etc/iptables/rules.v4 during install.
```

Step 3: Manage further changes - If you need to make any further rules changes then you can do that by running the appropriate iptables commands and then saving them out. Just keep in mind that since you added the DROP rule as the last rule in the chain you can't simply append new rules since they will be ignored. Instead, you must insert new rules before the final DROP rule. Here is a cookbook example of how you could add th opening a specified port after you have the initial rules set up.

Example: Adding a new opened port to the rules

```
# Open up a specified port by inserting the rule into the chain
sudo iptables -I INPUT 3 -p tcp --dport NNNN -j ACCEPT
       *** You must replace NNNN with the actual port number you want to open

# If you wanted to open up port a port to only the host with IP address 1.2.3.4 you could do that with:
sudo iptables -I INPUT 3 -s 1.2.3.4 -p tcp --dport NNNN -j ACCEPT

# Verify that the rule looks good and is in the right place
sudo iptables -v -L --line-numbers

# Save it out
sudo netfilter-persistent save
```

Here is another example showing how to remove an existing rule:

Example: Removing an open port from the rules

```
# View the current rules, with line numbers
sudo iptables -v -L --line-numbers

# Identify the rule you want to remove and note the line number.  Remove that line by number
sudo iptables -D INPUT N
        Note:  Replace 'N' with the line number of the rule to remove

# Verify that the rules looks good and the rule has been removed
sudo iptables -v -L --line-numbers

# Save it out
sudo netfilter-persistent save
```

11. **Block Brute-Force SSH Attacks** - If you have opened port 22 for ssh logins to the world in the firewall (which we strongly discourage) then you ***must*** take action to prevent brute-force login attempts. Furthermore, if you have enabled authentication using the IU ADS servers then this is doubly critical since leaving sshd open can result in users having their IU accounts blocked by repeated failed logins that ***will*** be generated by hacker bots. Here are the recommended options for securing ssh against such attacks.

    a. **Limit access in the firewall to just IU networks** - One simple and effective way to deal with this issue is to only open ssh port 22 in the firewall to IU networks (see previous section). If connections are needed from outside of IU then you can use the IU VPN.
    b. **Limit access in the firewall to IU networks plus a more limited set of non-IU networks** - If you know you need to be able to ssh in from a specific set of non-IU networks then you can just set things up so ssh is open to IU plus these extra subnets. You can use the *last* command to see where your connections are coming from and then open the firewall to these IP addresses or subnets.
    c. **Require ssh keys from outside of the IU network** - This is a really attractive option because it gives you unrestricted access from outside of IU but only using ssh keys. It is also extremely simple to set up by making a quick change to the sshd configuration as follows:

```
1) Set the following configuration parameter in /etc/ssh/sshd_config to prevent password
authentication:

    PasswordAuthentication no

2) At the end of /etc/ssh/sshd_config add this block to allow password authentication but only
from IU subnets:

    Match Address 129.79.*.*,156.56.*.*,149.159.*.*,149.160.*.*,149.161.*.*,140.182.*.*,149.165.*.
*,10.79.*.*,10.56.*.*
        PasswordAuthentication yes

3) Restart sshd

        sudo systemctl restart sshd
```

    Once this is set up, then you can use ssh keys from anywhere to connect. There are lots of resources on the web describing how to set up ssh keys so just google "ssh keys" plus your operating system and ssh software name for details (eg. google "windows putty ssh keys" or "linux openssh ssh keys")
    d. **Set up blocking software** - This is not a perfect option but if the other options are not feasible it does offer some protection. There are tools like Fail2ban that automatically adjust firewall rules dynamically to block hosts that generate too many failed logins. The problem with this approach is that most brute force ssh attacks are highly distributed so even a small number of failed logins that make it to the ADS server before an IP is blocked from a lot of different systems all add up to block accounts. Furthermore, this offers *ZERO* protection in the case where login credentials are compromised since there won't be any failed logins in that case. If you go this route and leave sshd open to the world you should also stop using ADS authentication as noted in the next step. Here are a couple good references on how to set it up:

        i. How to install Fail2ban on Ubuntu
        ii. How To Protect SSH and Apache Using Fail2Ban on Ubuntu Linux

    e. **Stop using ADS authentication** - If you absolutely must have sshd open to the world and the above options for restricting access or password-based authentication are not feasible for whatever reason then you should not use ADS authentication (per step Account Passwords above). But, if you go this route it is critical that you use local passwords that meet IU password requirements. Just keep in mind that this option offers *ZERO* protection in the case where login credentials are compromised which is why this approach is not recommended.
12. **Disable Guest Account** - There may be a guest account created by default so you should disable that per How do I disable the guest session?
13. **Disable Remote Root SSH Logins** - By default, Ubuntu doesn't allow any root logins. If you have changed that and are running sshd you must be sure that root logins are not allowed via remote ssh connections. You can ensure this is the case by making sure you have the line **PermitRootLogin no** in the sshd config files (/etc/ssh/sshd_config).

14. **Remove or Disable mDNS/avahi-daemon** - You should ensure that avahi-daemon is not running and providing mDNS services. This is almost certainly not needed on the IU network and can leave the system open to abuse. You can just remove the avahi-daemon package entirely (preferred) or disable it as follows:

```
Preferred: Remove the service

    sudo apt-get remove avahi-daemon

Alternate: Disable and stop the service

     echo manual | sudo tee /etc/init/avahi-daemon.override
     sudo stop avahi-daemon
```

15. **Securing Services** - You may need to run a variety of services on the system, including web and database servers. It is recommended that you limit networked services as much as possible and use IU and SICE servers if at all possible. Furthermore, if you do run such services it is best to limit their exposure on the network if they are only needed locally on the system. For example, a database server (like mysqld or mongod) that is used by a web server running on the same system need not be exposed on the network. We can't list every single service you may need to run but do have some specific recommendations for some common services in use within the school:

    - **sshd -** See the above section "Block Brute-Force SSH Attacks" for information about securing sshd against brute force attacks. Many distributions also enable insecure protocols so you should add the following 2 lines to the /etc/ssh/sshd_config file and restart the sshd service to correct this:

```
# Ubuntu 16.04
# ============
Ciphers chacha20-poly1305@openssh.com,aes128-ctr,aes192-ctr,aes256-ctr,aes128-gcm@openssh.com,
aes256-gcm@openssh.com,aes128-cbc,aes192-cbc,aes256-cbc
KexAlgorithms ecdh-sha2-nistp256,ecdh-sha2-nistp384,ecdh-sha2-nistp521,diffie-hellman-group-
exchange-sha256
```

    - **mongodb** - Mongodb is one of the most commonly misconfigured database applications. You are strongly encouraged not to expose this service on the network by using the **bind_ip=127.0.0.1** setting in your mongo config file. If you do modify that so it is exposed on the network, you **MUST** enable authentication with **auth=true** in your config. There are additional guidelines for securing mongodb in the MongoDB Security Checklist.
    - **mysqld/mariadb** - As with all other services, it is best not to expose them on the network, if possible. By default, most mysql installations will allow remote access by default but you should disable this if not needed. This can be done by adding the line **bind-address=127.0.0.1** to the **[mysqld]** section of the configuration file (usually /etc/my.cnf) and restarting the service. Most mysql installations also come with a script **mysql_secure_installation** that you should run to perform some basic security measures like setting a root password, removing default accounts and/or tables, and preventing remote root access.
    - **postfix** - In the vast majority of cases, you will want to configure the postfix email server to allow only local connection so you can send email. It is very unlikely that you will want to allow remote email connections so you should edit the /etc/postfix/main.cf configuration file and make sure you are using **inet_interfaces=localhost**.
    - **apache** - The apache web server is very popular and here are some basic configuration suggestions for making your installation more secure. This list is in no way exhaustive and there are lots of ways you can make your installation insecure. For that reason, you are encouraged to use central IU and SICE web services if at all possible.
        - SSL - If your site supports https/SSL you should disable SSLv1, SSLv2, and TLSv1 (aka. TLSv1.0) and only support TLSv1.1 and later. The following SSL configuration parameters are recommended:

```
SSLProtocol all -SSLv2 -SSLv3 -TLSv1
SSLCipherSuite HIGH:MEDIUM:!aNULL:!MD5:!RC4:!3DES:!IDEA
SSLHonorCipherOrder on
```

        - php.ini - Modify your /etc/php.ini and set **expose_php=Off**
        - TRACE - Disable TRACE request in your httpd.conf by adding "**TraceEnable Off**"
        - ETag Headers - Disable ETag headers in your httpd.conf by adding "**FileETag None**"
        - Directory Indexes - You should disable any automatics directory indexing that is not explicitly needed. To do that, look for any "Options" lines in your config files and remove the "Indexes" options from all of them that are not needed. In general, allowing directory indexes is a bad idea since you may inadvertently expose files you don't mean to expose.
    - **nginx** - Nginx is another web server that is widely used and here are some configuration tips. As with apache, there are lots of ways you can make your installation insecure. For that reason, you are encouraged to use central IU and SICE web services if at all possible.
        - SSL - If your site supports https/SSL you should disable SSLv1, SSLv2, and TLSv1 (aka. TLSv1.0) and only support TLSv1.1 and later. This can be done with the config option "`ssl_protocols TLSv1.1 TLSv1.2;`". You should also review the other security recommendations for SSL at Strong SSL Security on nginx.
        - General Recommendations - You are advised to check out the page How To Secure Nginx on Ubuntu 14.04 for some excellent tips on securing your installation.
    - **tomcat** - Here are some suggestions on securing tomcat:
        - SSL - If your site supports https/SSL, you should disable SSLv1, SSLv2, and TLSv1 (aka. TLSv1.0) and only support TLSv1.1 and later. There are different ways of doing this depending on the connector type you are using, but when using JSSE connectors you can use the following in the HTTPS connector configuration in the server.xml:

```
sslProtocol="TLS" sslEnabledProtocols="TLSv1.2,TLSv1.1"
```

- Manager App - Some tomcat installations will have a Manager App accessible via a url like http://host:port/manager/html or http://host:port/manager/. If this must be enabled, be sure to change the default password. Look for a file named tomcat_users.xml and modify the default usernames and passwords there. Look for both the admin and manager accounts and be sure to change both.
- General Recommendations - You are advised to check out the page Improving Apache Tomcat Security for some excellent tips on securing your installation.

16. **System Logging** - IU has specific logging requirements for all servers operating on the IU network per IT Policy IT-12. If your Ubuntu system is operating as a server on the IU network (eg. web, database, etc), you need to set up logging as follows:
    a. Auditd - You must set up auditd as described at Configuring and auditing Linux systems with Audit daemon. There is also information in the IU KB page INTERNAL (iu-kb): About Splunk audit configurations (you must log in to view this page). You can install this on Ubuntu with:

    ```
    sudo apt-get install auditd audispd-plugins
    ```

    Here is a minimal `/etc/audit/audit.rules` file that meets IT-12 requirements on a system with no sensitive data:

    ```
    -D
    -b 320
    -a exit,always -F arch=b64 -S open -F exit=-EACCES
    -a exit,always -F arch=b64 -S open -F exit=-EPERM
    -a exit,always -F arch=b32 -S open -F exit=-EACCES
    -a exit,always -F arch=b32 -S open -F exit=-EPERM
    ```

    This will log all failed file access attempts as well as both failed and successful logins. On a system with sensitive data, you must also log all successful file accesses. For example, if you were storing sensitive data in /home/goodies then you would add the following to the above auditd rules:

    ```
    -w /home/goodies -p wrxa
    ```

    Once you have the audit.rules file configured, you can restart auditd and verify by running:

    ```
    sudo /etc/init.d/auditd restart
    sudo auditctl -l
    ```

    Please let us know if you need any help setting this up.

    b. Splunkforwarder - The auditd data should be forwarded to the IU Log-Alert service using Splunkforwarder. This prevents possible tampering with the data that resides local to the system. The installation is done via an installer script that we can provide Please let us know and we can provide the installer script and packages.

17. **Account Maintenance** - You will need to review all user accounts on the system monthly and purge any accounts that are no longer needed.

18. **VMware Tools Installation** - If you are setting up a Ubuntu VM in the IU Intelligent Infrastructure (II) system you must install VMware tools. We recommend you use the open VMware tools as follows:

    ```
    sudo apt-get install open-vm-tools
    ```

19. **OS Patch Maintenance** - As noted above, you are required to configure the system so that security updates are installed automatically. However, you are encouraged to install other maintenance updates on a regular schedule.

20. **Security Vulnerability Scans** - All systems on the SICE networks will be automatically scanned using an external security scanner monthly. We will contact you to resolve any vulnerabilities that show up for your system and we expect that you will work promptly to resolve all reported issues.

21. **Breach Reporting** - In the event of a security breach, it must be reported immediately per IT Policy: Incident Response

If you have any questions about this or need further assistance, please contact us via the help desk.