# Introduction

While the early days of smartphones included tech giants such as Microsoft and Blackberry, today the Google Android and Apple iOS mobile operating systems account for virtually the entire global smartphone market. Despite Apple's early and arguably defining entrance into the smartphone market with the iPhone, the Android operating system accounts for the vast majority of global market share – 87% to Apple's 13% as of
2020
(https://www.statista.com/statistics/272307/market-share-forecast-for-smartphone-operating-systems).
Much of this success can be attributed to the open-source nature of Android applications versus Apple's "walled garden" approach.

Although the open-source security model has both advantages and disadvantages, Android devices are widely considered to be secure due to several built-in features. First, Android isolates applications within their own sandboxes, preventing other applications (including malware) from accessing data and other device features. They also employ granular permissions, which allow users to determine what level of access an application has (for example, camera, GPS, contacts, etc.). Google Play Protect provides an additional layer of security when downloading applications from Google Play and even when "sideloading" applications from third-party web sites.

Additional security measures can be applied should users choose. These include data encryption, multi-factor authentication, and application security settings. While use of these features is highly recommended, many users choose either very weak configurations or simply don't use them at all. Furthermore, careless browsing habits can lead to forms of attack known as clickjacking and drive-by downloads. Some users even attempt to circumvent security themselves by "rooting" the device to give them full administrative rights. As a result, most of the recent security issues have tended to be more user-based.

Device hardening, the process of applying all available security settings to a device, can ensure that a smartphone is protected against as many forms of compromise as possible. Hardening a smartphone includes configuring data encryption, applying multi-factor authentication, removing unnecessary applications and services, and verifying that all applications are using recommended security settings.

In this lab, you will be introduced to some of the basic security features of Android by way of virtual machines. Although several tools exist for this purpose, including Android Studio, this lab environment contains a stand-alone Android device. You will review and apply important security settings and application permissions. You will also install an app via the Google Play Store and verify the resulting security settings.

## Lab Overview

**SECTION 1** of this lab has two parts, which should be completed in the order specified.

1. In the first part of the lab, you will review basic Android security settings and permissions.

2. In the second part of the lab, you install an app from the Google Play Store and verify its permissions.

**SECTION 2** of this lab allows you to apply what you learned in **SECTION 1** with less guidance and different deliverables, as well as some expanded tasks and alternative methods. You will explore the relationship between Android and Linux using the Termux app. You will also compare the process of traditional app installation versus the process of downloading and installing an app from a third-party web site, a process known as “sideloading.”

Finally, you will explore the virtual environment on your own in **SECTION 3** of this lab to answer a set of questions and challenges that allow you to use the skills you learned in the lab to conduct independent, unguided work - similar to what you will encounter in a real-world situation.

## Learning Objectives

Upon completing this lab, you will be able to:

1. Understand the relationship between the Linux and Android operating systems.

2. Implement native security controls on the Android operating system.

3. Audit app permissions on the Android operating system.

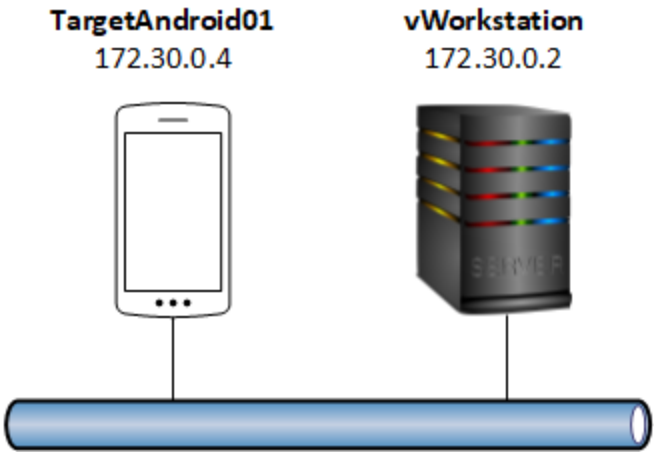
4. Install apps from the Google Play Store and third-party sources.

5. Run Linux commands on an Android device.

## Topology

This lab contains the following virtual machines. Please refer to the network topology diagram below.

- TargetAndroid01 (Android v9)
- vWorkstation (Windows: Server 2019)




## Tools and Software

The following software and/or utilities are required to complete this lab. Students are encouraged to explore the Internet to learn more about the products and tools used in this lab.

- Android Settings
- Google Play Store
- Termux
- BlueStacks

## Deliverables

Upon completion of this lab, you are required to provide the following deliverables to your instructor:

**SECTION 1**

1. Lab Report file, including screen captures of the following:

- Results of the Google Play Protect scan
- Updated “last successful check for update” timestamp
- Android lock screen
- List of apps that currently have Camera permissions
- List of apps that currently have Location permissions
- List of apps that currently have Microphone permissions
- App details for Chess Free
- Updated Ads settings

2. Any additional information as directed by the lab:

- None

**SECTION 2**

1. Lab Report file, including screen captures of the following:

- Encryption set-up explanation
- Find My Device settings
- Display settings with the updated auto-lock timer
- Deactivated USB debugging setting
- Firefox Nightly app details
- Deactivated Install unknown apps setting for Chrome

2. Any additional information as directed by the lab:

- None

**SECTION 3**

1. Lab Report file, including screen captures of the following:

- Results of the cat /proc/cpuinfo command
- Results of the cat meminfo command

2. Any additional information as directed by the lab:

- Firefox Nightly app running in BlueStacks

# Section 1: Hands-On Demonstration

**Note:** In this section of the lab, you will follow a step-by-step walk-through of the objectives for this lab to produce the expected deliverables.

1. **Review** the **Tutorial**.

   Frequently performed tasks, such as making screen captures and downloading your Lab Report, are explained in the Cloud Lab Tutorial. The Cloud Lab Tutorial is available from the User menu in the upper-right corner of the Student Dashboard. You should review these tasks before starting the lab.

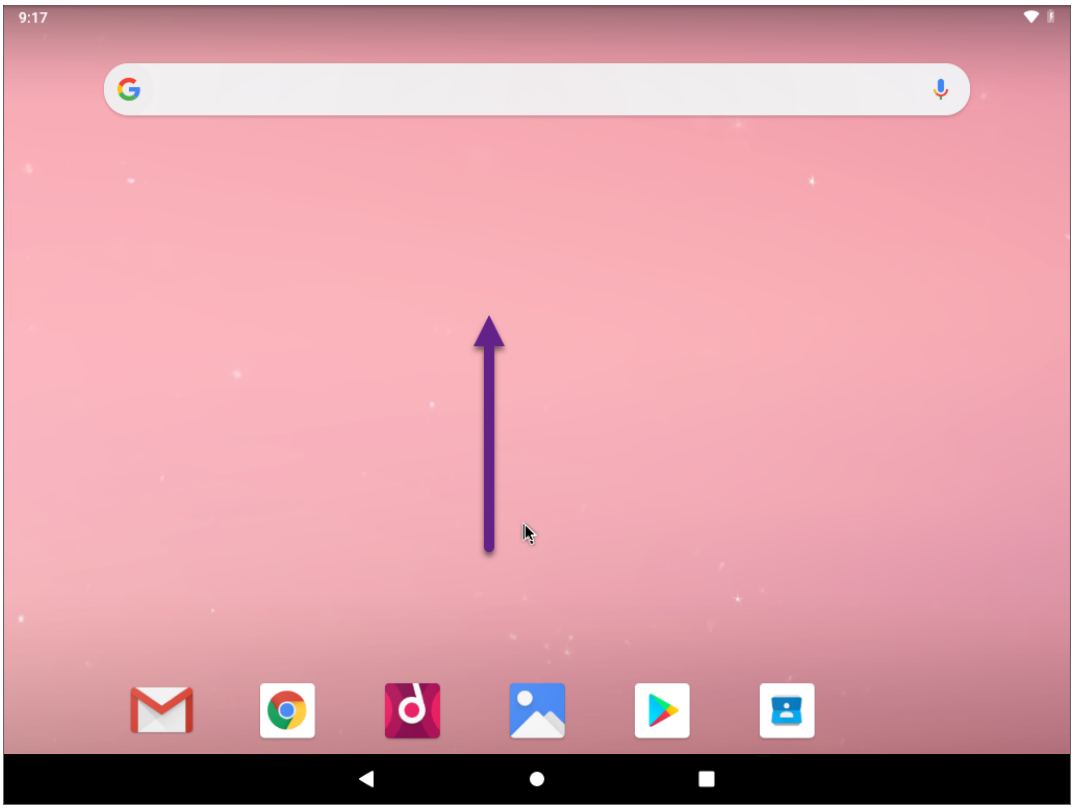
2. **Proceed** with **Part 1**.

## Part 1: Apply Basic Android Security Controls

**Note:** The Android operating system is based on the open-source Linux operating system. Linux is a powerful, stable, and robust platform, and it is found running everything from desktop PCs to large data center servers. It is also flexible enough to run on small devices, such as home Wi-Fi routers, IoT devices and, yes, even phones. While the Android operating system benefits from many of the built-in security controls offered by Linux, including process sandboxing, strict user privileges, and the use of the Security-Enhanced Linux (SELinux) module, it also provides users with the freedom and flexibility to degrade those controls. When targeting Android devices, many attackers will exploit missing or weak security configurations introduced by users who either lack awareness or perceive the security settings as inconvenient. Meanwhile, deliberate attempts by savvy users to gain administrative access by "rooting" the device or sideload potentially dangerous apps from third-party distributors further complicate the Android security ecosystem.

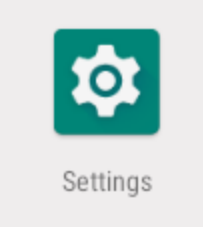
In this part of the lab, you will explore several best practices for improving the security posture of an Android device, including enabling Google Play Protect, regularly checking for security updates, using screen lock, and monitoring app permissions.

1. At the TargetAndroid01 home screen, **position your cursor** in the middle of the display’s background, then **click and drag up** to simulate the ‘swipe up’ action and open the apps screen.



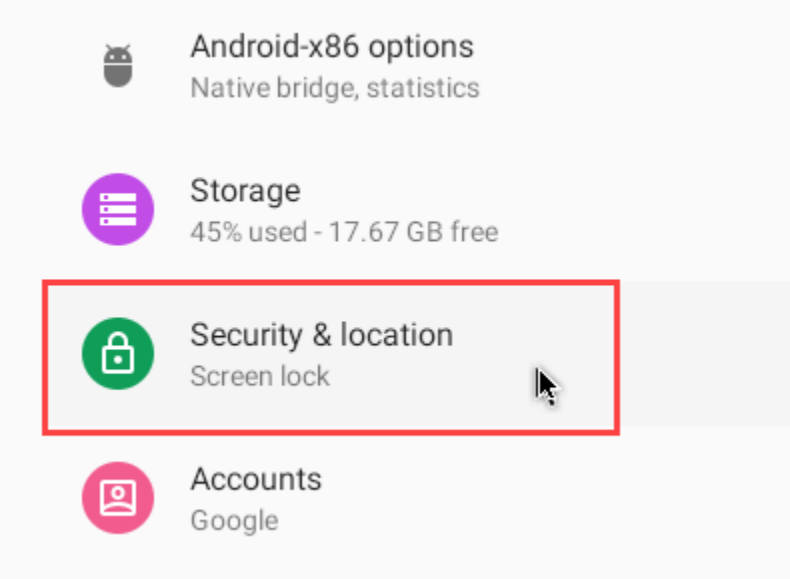

Home screen

2. At the Apps screen, **click** the **Settings icon** to open the Settings app.




Settings icon

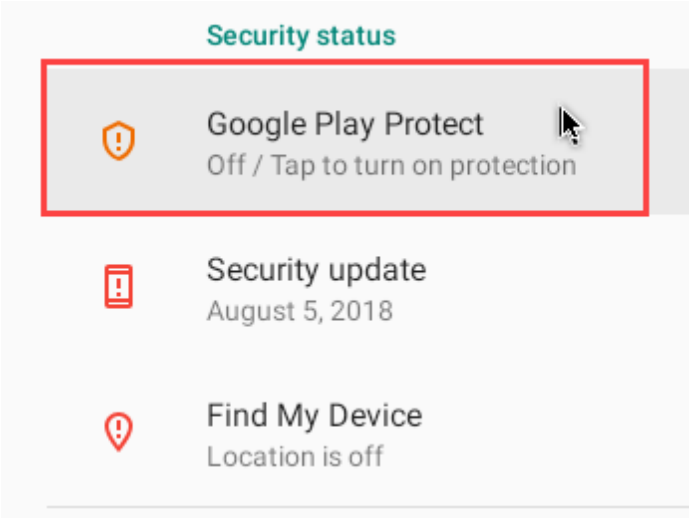
3. In the Settings app, **click Security & location** to open the Security & location settings.




Security & location settings

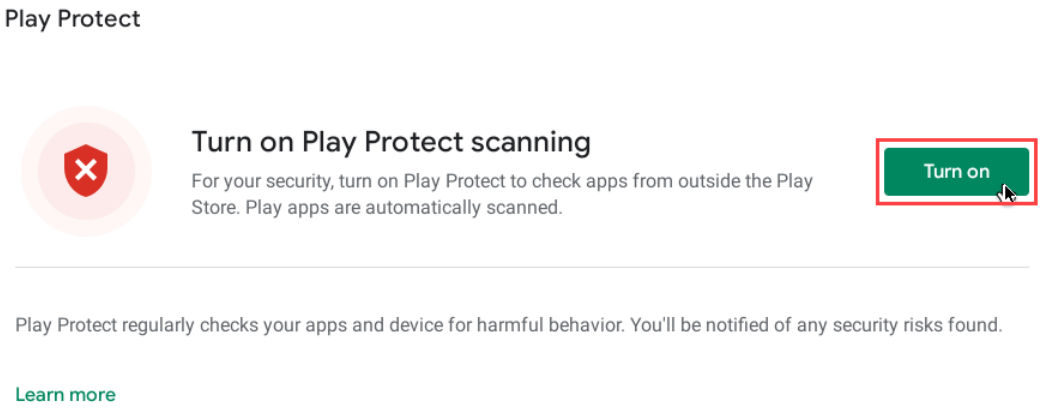
**Note:** You should see three settings under the Security status section: Google Play Protect, Security update, and Find My Device. Google Play Protect provides anti-malware protection for all installed apps, including apps that have been downloaded outside of the Google Play Store. Security update refers to the latest security update that has been applied to this device. Find My Device contains the settings related to Android's Find My Device functionality. In the next steps, you will use Google Play Protect to run a malware scan.

4. In the Security status section, **click Google Play Protect** to open the Google Play Protect settings.




Google Play Protect

5. On the Play Protect page, **click** the **Turn on button** to activate Google Play Protect.




Turn on Google Play Protect

**Note:** Play Protect will automatically start a malware scan, which should finish after just a few seconds and confirm that “no harmful apps were found.” Once Play Protect is activated, you can re-run a scan at any time by returning to this screen and clicking the Scan button.

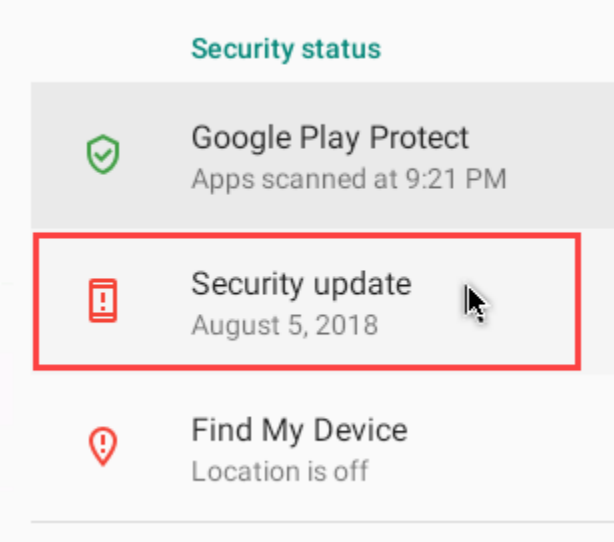
6. **Make a screen capture** showing the **results of the Google Play Protect scan**.

7. At the bottom of the screen, **click** the **Triangle icon** to return to the Security & location settings.

Triangle icon

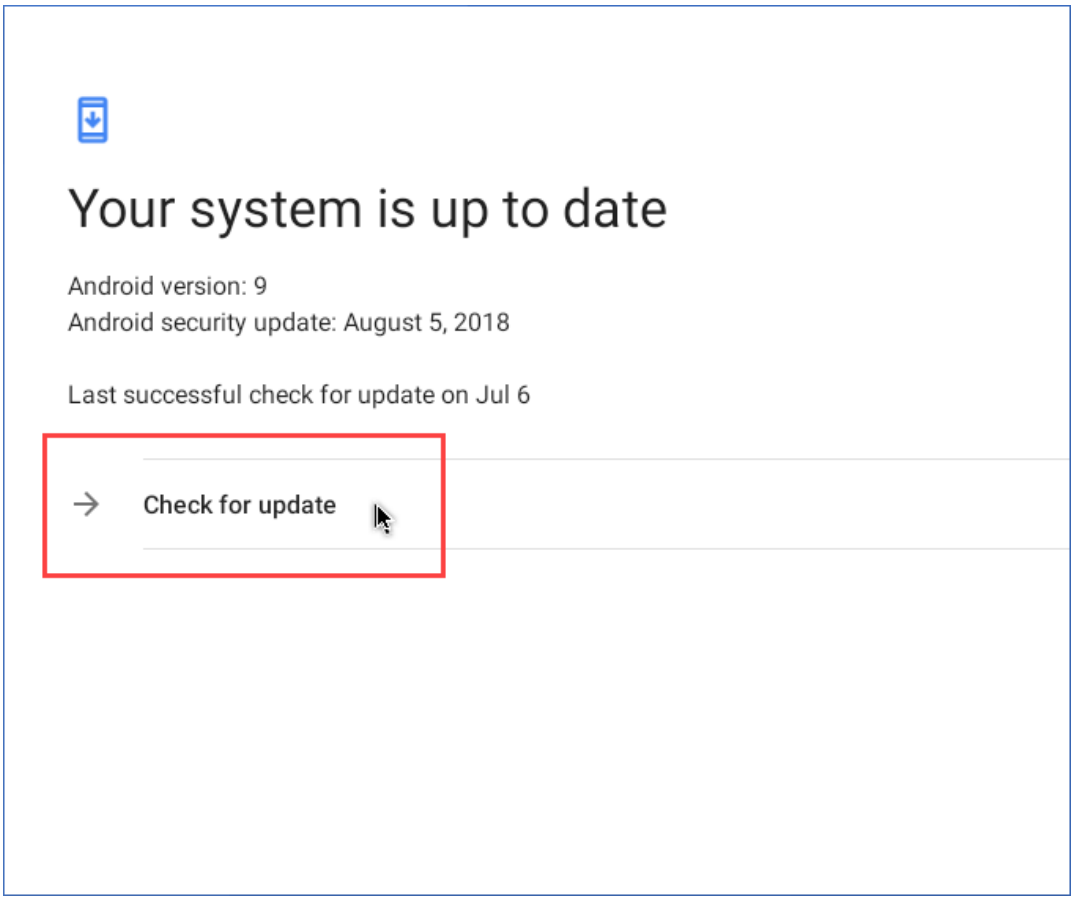
**Note:** Installing regular security updates is a critical part of keeping any system secure, and smartphones are no different. In the next steps, you will review the security update settings for this device and manually check for new updates.

8. In the Security status section, **click Security update** to open the Security update settings.



Security update settings

9. On the Security update page, **click Check for update** to query Google's servers for the latest security patch.



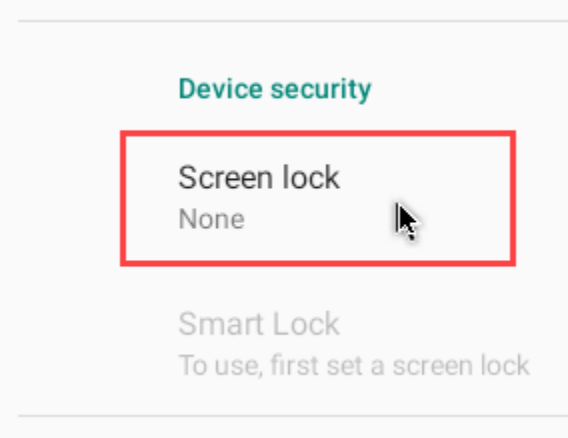

Check for update

**Note:** A message will display stating that "Your system is up to date." This is not entirely true. While this version of Android (Version 9) is several years old, Google committed to releasing security patches for Android Version 9 through Fall 2021, and the latest security update appears to be from August 2018. This is a limitation of running a mobile operating system in a virtualized environment and may be disregarded. In practice, you should ensure your smartphone always has access to the latest security updates. While it may be fair to wait a few months to update to major OS releases, knowing that they can sometimes be buggy at launch, you should never wait to install a security update.

10. **Make a screen capture** showing the **updated "last successful check for update" timestamp**.

11. At the bottom of the screen, **click** the **Triangle icon** to return to the Security & location page

12. In the Device security section, **click Screen lock** to open the Screen lock settings.

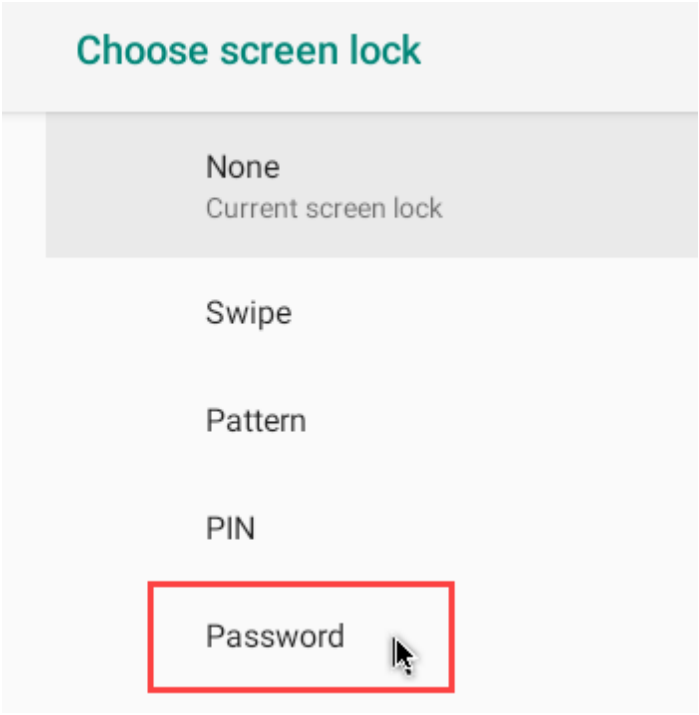



Screen lock

**Note:** The different screen lock settings include None, Swipe, Pattern, PIN and password. Some devices also support biometric features such as fingerprint and facial recognition. None and Swipe offer no actual protection. Pattern, a medium level of security, allows the user to draw a sequence of dots on the display (at least four dots must be connected to complete a pattern). PIN requires the use of a four-digit or higher number and is considered higher security than Pattern. Password requires a combination of at least four numbers and/or letters and is considered the most secure option. Some smartphone manufacturers might argue that their cutting-edge biometric authentication features are more secure, and while this may be technically correct in terms of the unique complexity of the key, the fact of that matter is it is much easier for someone to hold your phone up to your face than it is for them to force you to disclose your password. As usual, multi-factor authentication schemes are preferable, but should be balanced against convenience and context.

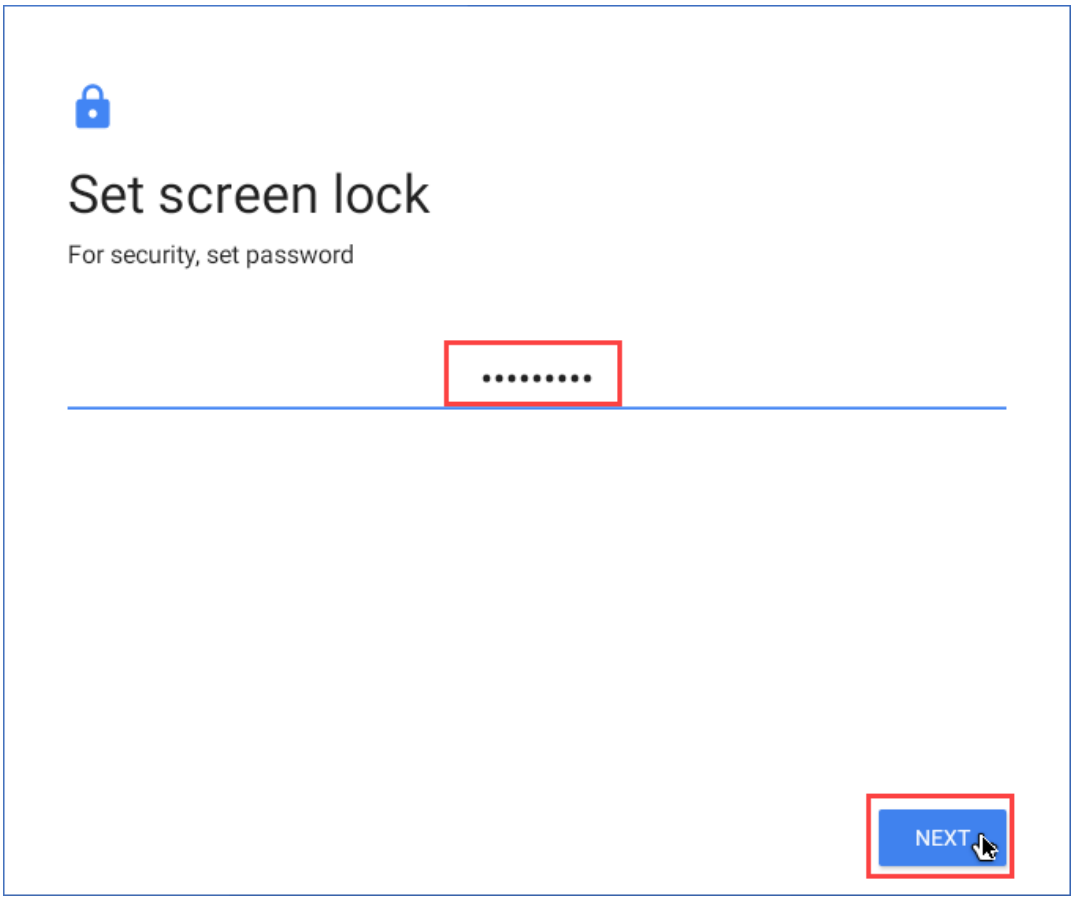
In the next steps, you will configure the Android device to use the Password screen lock. While not the most secure, it will add more protection than the device currently has, and it will be the simplest to implement without an actual touchscreen.

13. On the Choose screen lock page, **click Password** to open the Set screen lock page.



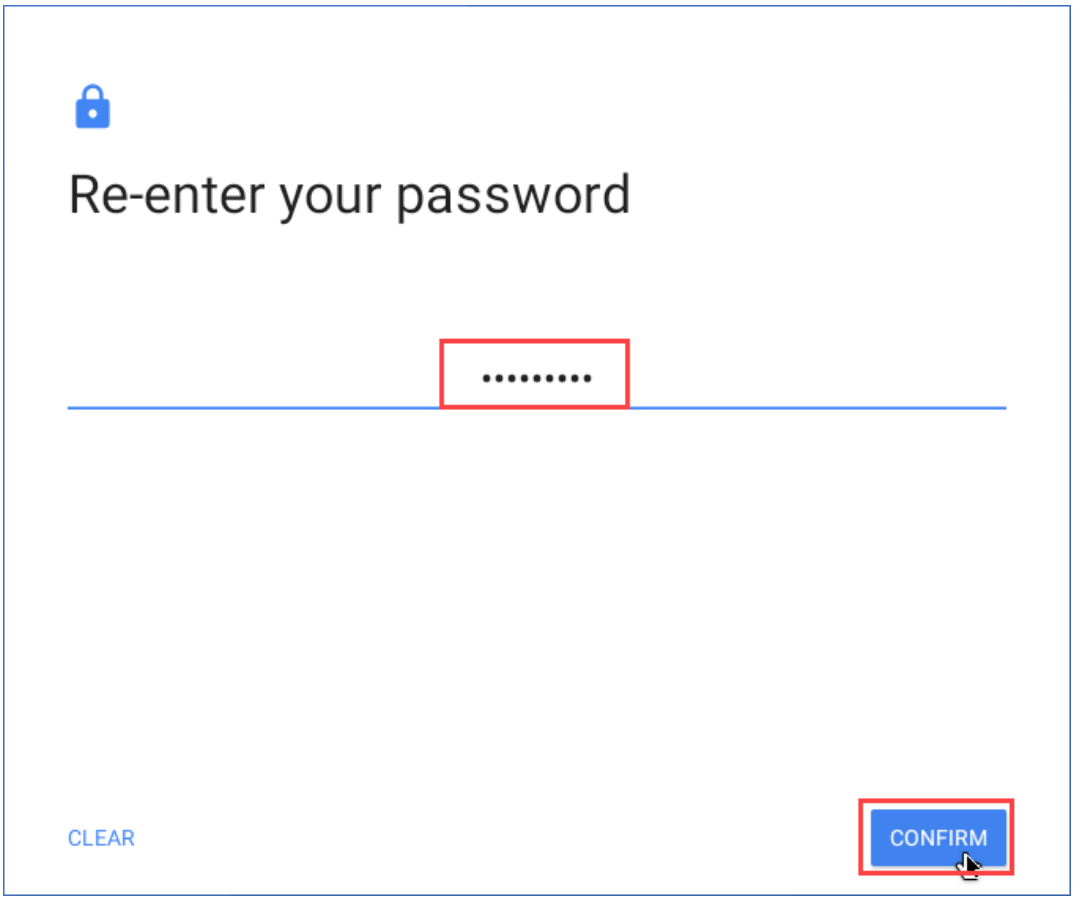

Password option

14. On the Set screen lock page, **type** `P@ssw0rd!`, then **click Next** to continue.



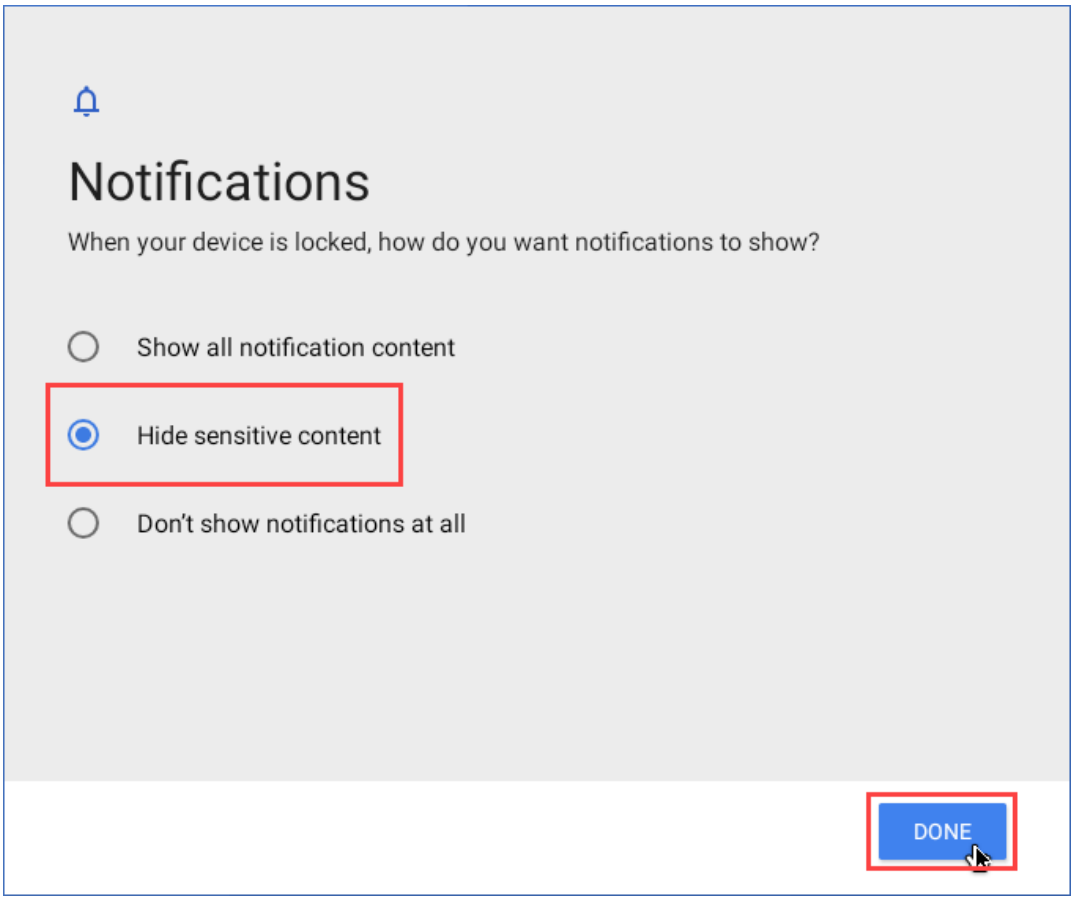

Set screen lock

15. On the Re-enter your password page, **type** `P@ssw0rd!` again, then **click Confirm** to continue.



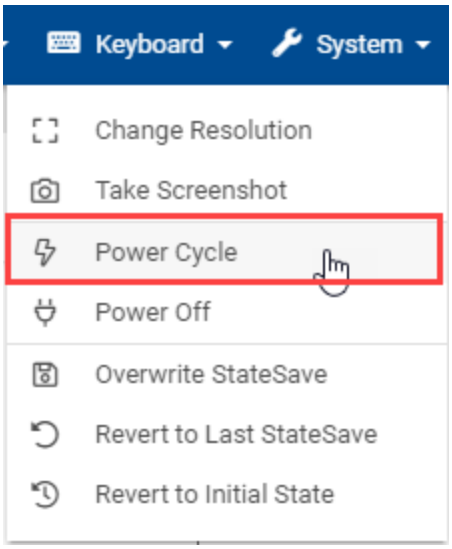

Re-enter your password

16. On the Notifications page, **click** the **Hide sensitive content radio button**, then **click Done** to continue.




Notifications

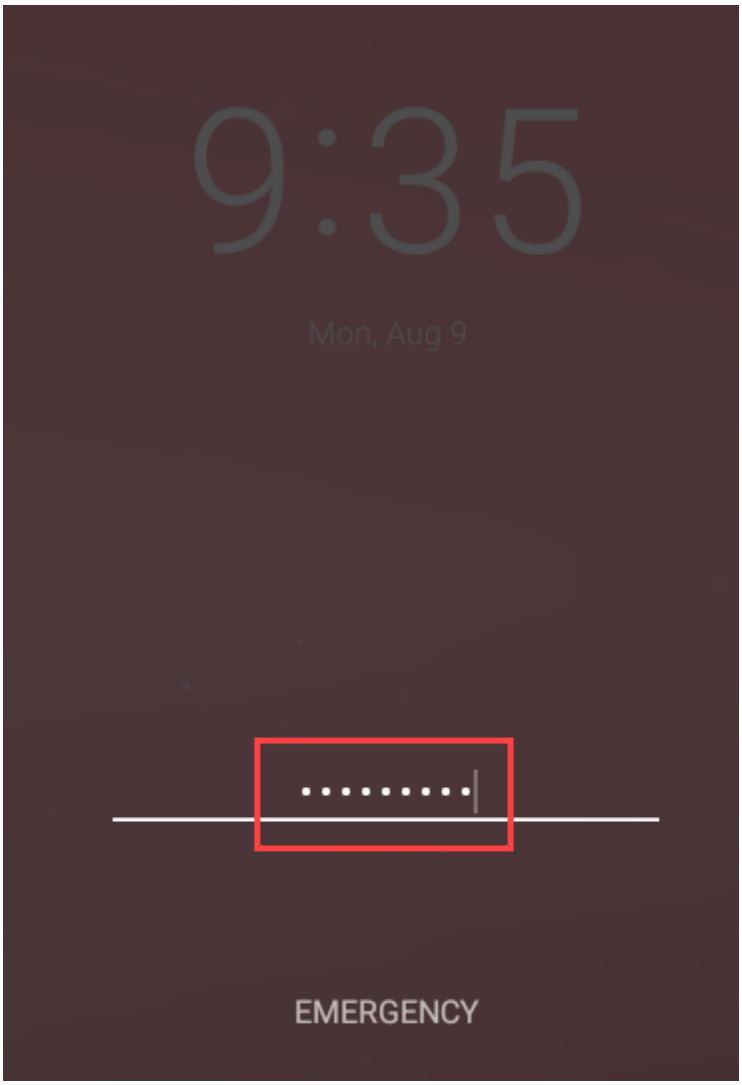
17. On the Lab View toolbar, **select Power Cycle** from the System menu to restart the Android VM.




System > Power Cycle

18. **Make a screen capture** showing the **Android lock screen**.

19. **Press** the **space bar** to simulate the swipe up gesture, then **type** `P@ssw0rd!` and **press Enter** to unlock the smartphone.

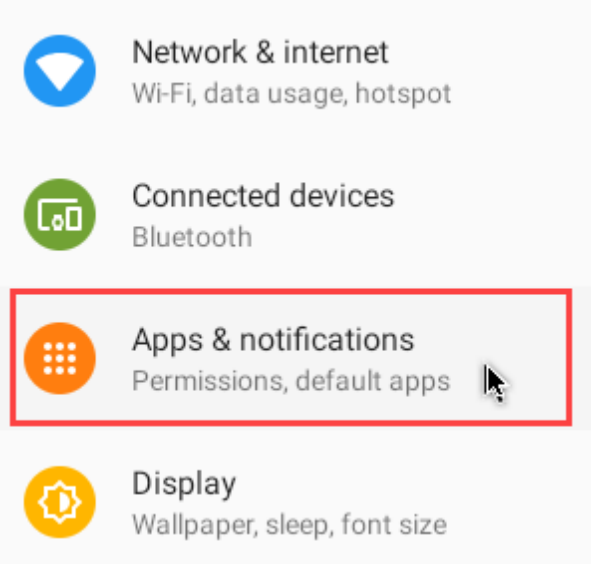



Log-in screen

20. **Repeat steps 2-3** to return to the Settings app.

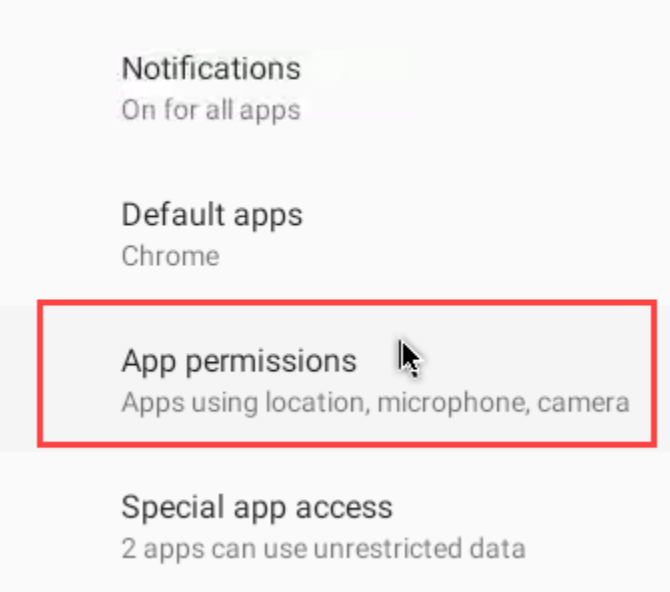
**Note:** In the next steps, you will conduct a brief audit of the most sensitive app permissions. While earlier smartphones lacked this capability, modern versions of both Android and iOS allow users to assign granular access controls to individual apps for each of their device's major functions, such the camera, microphone, location services, and even user data. As a rule, you should always limit each app to the minimum level of permissions required for its use.

21. In the Settings app, **click Apps & notifications** to open the Apps & notifications settings.



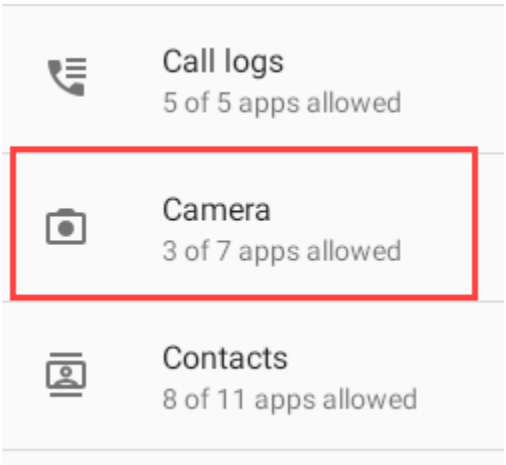

Apps & notifications

22. On the Apps & notifications page, **click App permissions** to open the App permissions page.



App permissions

23. On the App permissions page, **click Camera** to open the Camera permissions page.




Camera permissions

24. **Make a screen capture** showing the **list of apps that currently have Camera permissions**.

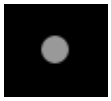
25. **Click** the **back button** to return to the App permissions page.

26. **Repeat steps 23-25** for Location and Microphone.

27. **Make a screen capture** showing the **list of apps that currently have Location permissions**.

28. **Make a screen capture** showing the **list of apps that currently have Microphone permissions**.

29. At the bottom of the screen, **click** the **Circle icon** to return to the home screen.



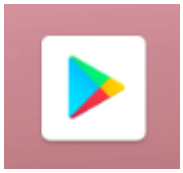
Circle icon

## Part 2: Install an App and Verify Privacy Settings

**Note:** In this part of the lab, you will install a new app from the Google Play Store and examine its permissions. While the Google Play Store and Apple App Store both position themselves as a secure marketplace of trusted developers, the reality is that malicious apps still routinely make their way into wide circulation through these channels. Even among technically legitimate apps that do not rise to the level of malware, there are many that default to overly permissive privacy and permissions settings, potentially exposing user data to bad actors (or just sophisticated advertising companies, much like Google itself).

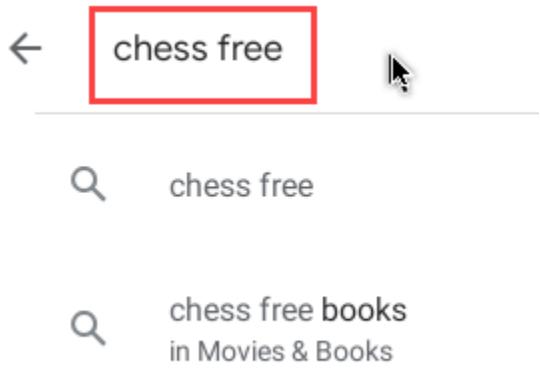
In the next steps, you will access the Google Play Store to install an app.

1. At the TargetAndroid01 home screen, **click** the **Play Store icon** to open the Play Store app.



Play Store icon

2. In the Play Store app, **type** `chess free` in the Search for apps & games field, then **press Enter** to run the search.




Search field

3. In the search results, **click** the **Chess Free icon** (by AI Factory Limited) to open the Chess Free app page.

4. On the Chess Free app page, **click** the **Install button** to download the app.

**Note:** As the app is installing, you should see a message stating that the app is "Verified by Play Protect".

5. When the app is finished installing, **click** the **Play button** to launch the app.


Play button

**Note:** Pay attention to the message that appears regarding the app's advertising and privacy policy.

While the prospect of reading a full Terms & Conditions statement can be daunting, it is important to understand what sort of terms and conditions these policies contain. Even among large, publicly traded companies, it is an industry standard practice to include some form of language authorizing the collection and use of personal information within an app for business intelligence and analytics. When installing free apps from unknown developers in the Play or App Store, these policies can include much stronger language that authorizes the developer to scrape and share personal information directly from your smartphone. Even when there may not be a formal “I Accept” button, using the app can constitute an implicit acceptance of the agreement.

In the next steps, you will verify the app’s current privacy and permissions settings.

6. **Click** the **OK button** to close the privacy notification.

7. At the bottom of the screen, **click** the **Circle icon** to return to the home screen.

8. At the TargetAndroid01 home screen, **position your cursor** in the middle of the display’s background, then **click and drag up** to simulate the "swipe up" action and open the Apps screen.

9. At the Apps screen, **click** the **Settings icon** to open the Settings app.

10. In the Settings app, **click Apps & notifications** to open the Apps & notifications settings.

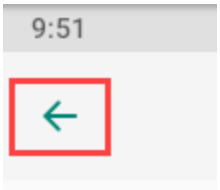
11. In the Recently opened apps section, **click** the **Chess Free icon** to view the app page.

**Note:** Due to the limitations of virtualizing Android, the Chess Free icon was selected for its minimal system load, rather than as a case study in aggressive over-permissioning. There's not much to see here, but if the Chess Free app did request specific permissions, you would be able to view and manage them here.

In the final steps, you will examine the Google Ads setting, which allows you to disable the personalization of advertising content.

12. **Make a screen capture** showing the **app details for Chess Free**.

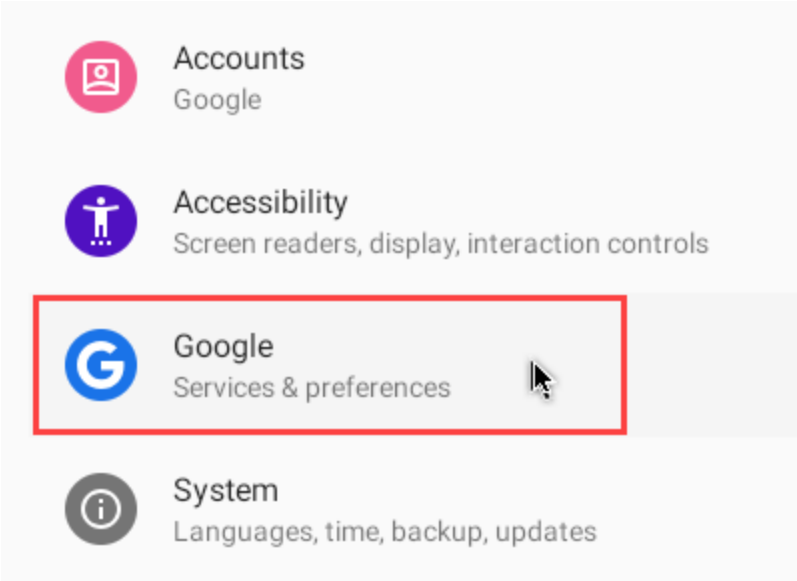
13. **Click** the **back button** to return to the Apps & notifications page.




Back button

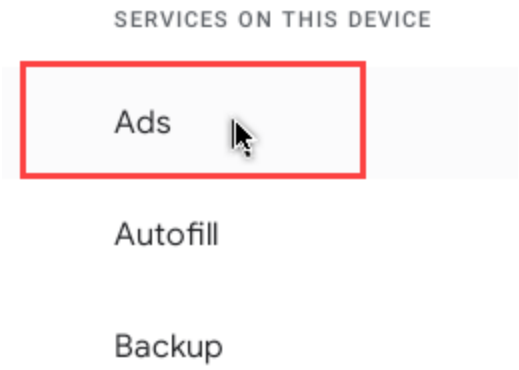
14. **Click** the **back button** again to return to the Settings page.

15. On the Settings page, **click Google** to open the Google services & preferences settings.



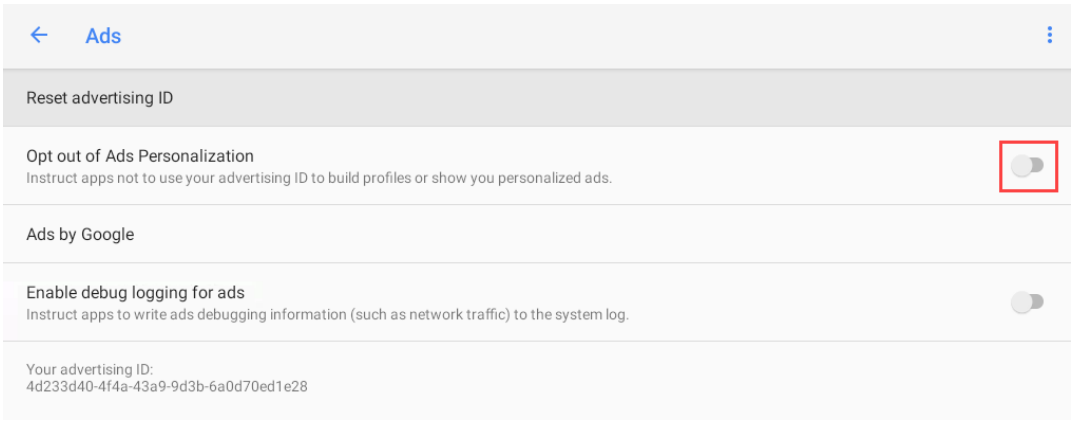

Google services & preferences

16. In the Services on this device section, **click Ads** to open the Ads page.




Ads

17. On the Ads page, **click** the **Opt Out of Ads Personalization switch** to activate this setting.




Opt Out of Ads Personalization

18. When prompted, **click OK** to continue.

**Note:** As they say, there's no such thing as a free lunch. While many of Google's services appear to

be free, the fact of the matter is that they are making money from their users by selling advertising services to other businesses. One way that they accomplish this is through the use of an anonymized unique identifier that is automatically assigned to each user in the Google ecosystem. By activating this Opt Out feature, you are effectively blocking apps from using your advertising ID to track you and display personalized ads.

19. **Make a screen capture** showing the **updated Ads settings**.

20. At the bottom of the screen, **click** the **Circle icon** to return to the home screen.

**Note:** This concludes Section 1 of the lab.

# Section 2: Applied Learning

**Note: SECTION 2** of this lab allows you to apply what you learned in **SECTION 1** with less guidance and different deliverables, as well as some expanded tasks and alternative methods. You will apply additional security controls to the virtual Android device.

## Part 1: Apply Additional Android Security Controls

**Note:** In Section 1 of the lab, you applied several basic security controls to harden the virtual Android device. Depending on your risk tolerance as an average smartphone user, Play Protect, regular security updates, a lock screen, and a general understanding of app permissions will together provide a fairly strong security posture – or at least one that is not needlessly vulnerable. However, smartphone usage is not limited to personal use and personal risk – even when you own the phone. Driven by the rise in remote work and ever-increasing smartphone sophistication, many employers have adopted Bring Your Own Device (BYOD) policies that allow employees to use their own devices for work-related tasks. If you are using your smartphone for work, even just email and chat, your company's security perimeter now extends to your smartphone. For this reason, many organizations with BYOD policies will require or at least strongly recommend additional hardening measures for smartphones.

In this part of the lab you will apply several additional security controls, including enabling device encryption, updating the auto-lock timer, disabling Developer Options, and enabling Device Manager.

1. If necessary, **unlock** the **TargetAndroid01 device**.

2. From the TargetAndroid01 home screen, **open** the **Settings app** and **navigate** to the **Security & location settings.**

3. In the Security & location settings, **expand** the **Advanced category**, then **open** the **Encryption & credentials settings**.

4. In the Encryption & credentials settings, **click Encrypt tablet** to enable encryption on this device.

**Note:** At this point, you will be prompted with an explanation of the Encrypt feature, followed by a prompt to "Charge your battery and try again." This is yet another restriction of the virtualized environment, so you will not be able to proceed any further. On a real Android device, enabling encryption is one of the strongest security measures you can apply. Without encryption, if your

smartphone should fall into the wrong hands, those wrong hands could easily access your data with the right software. That said, even with encryption enabled, sophisticated, well-resourced, and determined actors (government agencies, security companies, and criminal hacking groups) can still bypass that encryption, but the software used in those situations is typically weapons-grade and highly unavailable to the average person.

5. **Make a screen capture** showing the **encryption set-up explanation**.

**Note:** In the next steps, you will review the Find My Device feature.

6. **Return** to the **Security & location settings**, then **click Find My Device**.

**Note:** Once again, Find My Device is a feature that cannot be activated due to the restrictions of the virtualized environment. On a real Android device, Find My Device (also known as Android Device Manager) is a powerful security feature that allows you to remotely track, lock, and wipe your Android device. Naturally, wiping your device is a last-resort measure, but in the event that you should lose your phone - or worse, have it stolen - having this feature enabled can provide peace of mind for both you and your employer that any sensitive data will not be easily accessed.

7. **Make a screen capture** showing the **Find My Device settings**.

**Note:** In the next steps, you will update the Android auto-lock (or Sleep) settings.

8. **Return** to the main **Settings page**, then **open** the **Display settings**.

9. In the Display settings, **click Sleep** to open the Sleep timer options.

10. From the Sleep options, **select 2 minutes**.

**Note:** Although this hardening measure is relatively simple compared to Device Encryption and Find My Android, it is an important complement to the Screen Lock you applied in Section 1. Currently, the

TargetAndroid01 device is set to sleep (and by extension, lock itself) after 10 minutes of inactivity. While a 2-minute and a 10-minute window may be functionally equivalent if someone grabs your phone directly from your hands, it can make all the difference if you set your phone down unattended and someone sees an opportunity to stroll by and leisurely drop it in their pocket.

11. **Make a screen capture** showing the **Display settings with the updated auto-lock timer**.

12. **Change** the Sleep settings back to **10 minutes**.

**Note:** Although a shorter Sleep timer makes for better security, it can get rather annoying when working on a lab.

In the next steps, you will review Android's developer options.

13. **Return** to the main **Settings page**, then **open** the **System settings**.

14. In the System settings, **expand** the **Advanced category**, then **click Developer options** to open the list of Developer options settings.

**Note:** Although not inherently dangerous, the Developer options contain certain settings that can weaken your device's security posture if activated. One of the most notable is the USB debugging option, which allows a developer to fully control a device using a USB connection to a separate computer.

Although normally hidden, the Developer options can be revealed by navigating to the About settings, then clicking the Build number seven times.

15. Review the Developer options page to **verify** that **USB debugging** is **deactivated**.

16. **Make a screen capture** showing the **deactivated USB debugging setting**.

17. **Return** to the **home screen**.

## Part 2: Test and Disable App Sideloading

**Note:** Android apps are distributed as Android Packages, or APK files. As demonstrated in Section 1, the normal process for installing apps using the Google Play Store is fairly straightforward. However, there are certain situations when an app is not available through the Play Store, but the user still wishes to install it on their device. For example, a company may be developing a new app strictly for internal use, in which case they would have no reason to distribute the app through the Play Store. This process, known as sideloading, is fairly simple to perform on Android devices, but should be handled with great caution. Some apps may contain potentially unwanted applications (PUAs); enable permissions settings that could expose user data, contacts, browsing history, and other sensitive information; or simply be malware. Although Google Play Protect provides anti-malware protection for all apps installed on the device, including apps that are sideloaded, as a best practice, third-party app downloads should remain disabled until a specific need from a trusted source arises.

In this part of the lab, you will temporarily update the appropriate settings on the TargetAndroid01 device to permit sideloading. You will then download a trusted third-party app from a trusted third-party source. Finally, you will restore the original settings to disable further third-party downloads.

1. From the TargetAndroid01 home screen, **open** the **Settings app** and **navigate** to the **Apps & notifications settings**.

2. In the Apps & notification settings, **expand** the **Advanced category**, then **click Special app access**.

3. In the Special app access settings, **select Install unknown apps** and **verify** that **Chrome** is set to **Allowed**.

4. **Return** to the **home screen**.

5. From the home screen, **launch** the **Chrome app**.

6. In the Chrome app, **navigate** to **`www.apkmirror.com`**.

**Note:** There are several sites available for downloading apk files, some safer than others. APKmirror.com is owned and operated by the same company as a widely-read Android news and reviews website (androidpolice.com) and is regarded as a trustworthy third-party source of APK files. In the next steps, you will install Firefox Nightly for Developers, an experimental (and potentially unstable) build of the popular Firefox browser.

7. On APKmirror.com, **search for** and **install** the APK file for the latest x86 version of the **Firefox Browser (Nightly for Developers)** app, following the installation prompts as they appear.

8. When the installation is complete, **open Firefox Nightly**, then **return** to the **home screen**.

9. From the TargetAndroid01 home screen, **open** the **Settings app** and **navigate** to the **Apps & notifications settings**.

10. In the Recently opened apps section, **click Firefox Nightly** to display the app permission settings.

**Note:** If you must sideload an app on your Android device, be sure to follow the same permissions audit that you would conduct with a normal app.

11. **Make a screen capture** showing the **Firefox Nightly app details**.

12. **Return** to the **Install unknown apps settings**, then **deactivate** this setting for **Chrome.**

13. **Make a screen capture** showing the **deactivated Install unknown apps setting for Chrome**.

14. **Return** to the **home screen**.

**Note:** This concludes Section 2 of the lab.

# Section 3: Challenge and Analysis

**Note:** The following exercises are provided to allow independent, unguided work using the skills you learned earlier in this lab - similar to what you would encounter in a real-world situation.

## Part 1: Gather System Specs with Linux Commands

Acme, Inc is preparing to deploy an in-house app for all employee mobile devices. The app will be made available via the company's internal web site and will not be available on the Google Play Store. As a member of the cross-functional DevSecOps team responsible for developing the new app, your manager has asked you to assist with the testing process. As a first step, your manager has asked you to gather some detailed system specifications from the Android tablet that the team has been using for testing. Fortunately, you know that Android is a modified version of Linux and that you will be able to use Linux commands to gather that required information.

From the TargetAndroid01 device, open the Terminal Emulator app. Terminal Emulator is an Android app that provides a command line interface for executing Linux commands.

At the command prompt, **execute** `cat /proc/cpuinfo` to display information about the CPU.

**Make a screen capture** showing the **results of the cat /proc/cpuinfo command**.

At the command prompt, **execute** `cat /proc/meminfo` to display information about the system memory.

**Make a screen capture** showing the **results of the cat /proc/meminfo command**.

## Part 2: Run an Android App on Windows with BlueStacks

In this lab, you worked with the Android OS on a virtual device, closely approximating the behavior of an actual phone experience. However, when developers want to test a new app, they typically prefer to work in dedicated development environments such as Android Studio or similar products. In this case, your company uses a product called BlueStacks, which provides Android device emulation for Windows and Mac OS systems. Your manager has requested you test the app using BlueStacks to verify that it loads correctly.

From the Lab View toolbar, select the vWorkstation from the Virtual Machine menu, then log in using the password P@ssw0rd!. Launch the BlueStacks application from the vWorkstation. Next, visit APKmirror.com and download the x86 version of the apk file for Firefox Browser (Nightly for Developers). Once the download is complete, open the Firefox app in BlueStacks.

**Make a screen capture** showing the **Firefox Nightly app running in BlueStacks**.

**Note:** This concludes Section 3 of the lab.