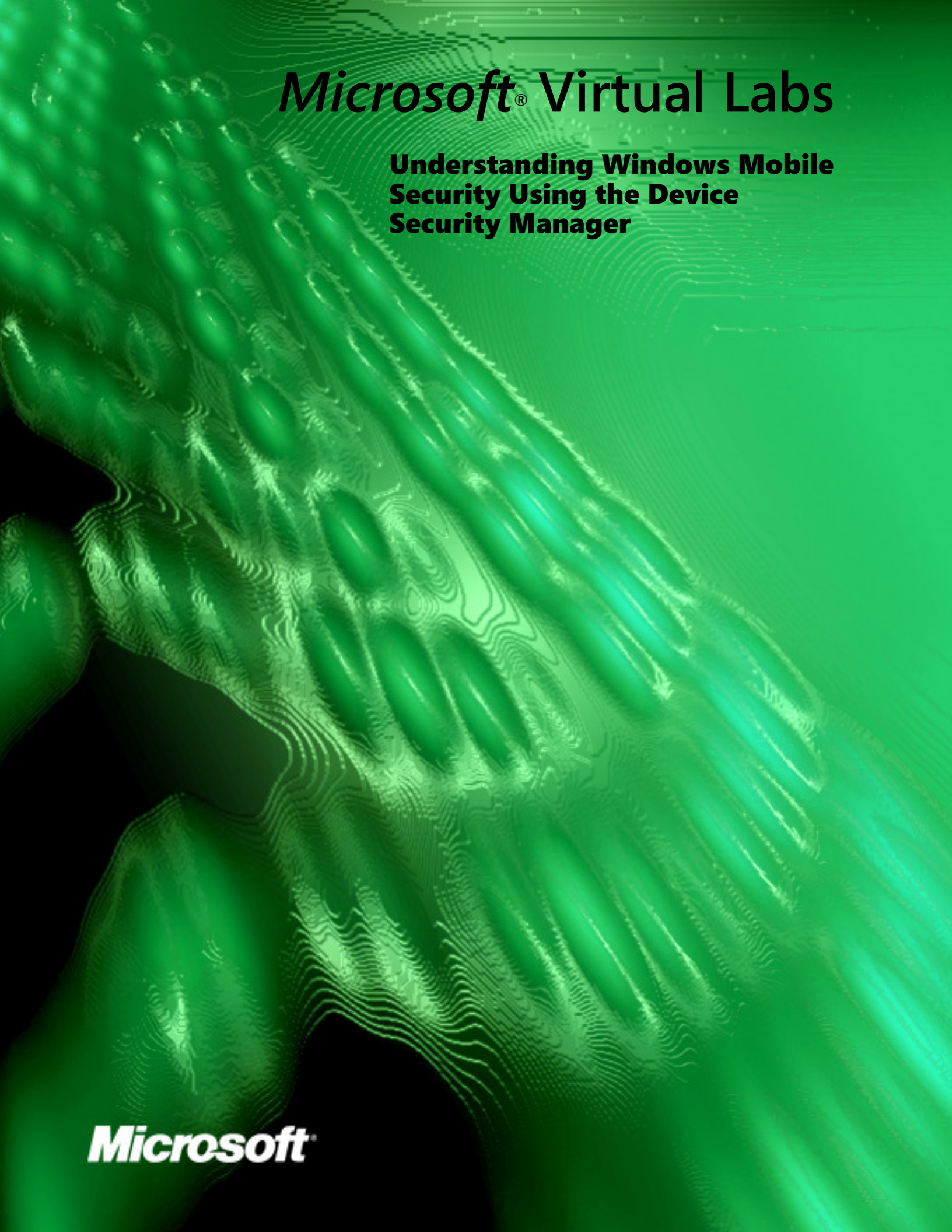


The background is a vibrant green with a complex, abstract pattern. On the left side, there is a dark, shadowed shape that resembles a hand or a stylized figure, possibly representing a user or a device. Overlaid on this and the rest of the background are intricate, glowing patterns that look like circuitry or data flow, with lines and nodes connecting different points. The overall effect is high-tech and digital.

Microsoft® Virtual Labs

**Understanding Windows Mobile
Security Using the Device
Security Manager**

Microsoft®

Table of Contents

Understanding Windows Mobile Security Using the Device Security Manager	1
Exercise 1 Using the Device Security Manager to Determine the Security Configuration of a Device	2
Exercise 2 Using the Device Security Manager to Change the Security Configuration of a Device.....	5
Exercise 3 Provisioning the Development Certificates on a Device.....	7
Exercise 4 Signing a File with the Development Certificates.....	10
Exercise 5 Provisioning a Locked Device	14
Exercise 6 Learning More about Mobile2Market.....	15
Summary.....	16

Understanding Windows Mobile Security Using the Device Security Manager

Objectives

The objective of this lab is for you to understand the Windows Mobile security model and the different security configurations. You will also learn how to use several tools to see the configuration on a given device, sign a file, and check a file signature.

In this HOL, you will perform the following exercises:

- Using the Device Security Manager to determine the security configuration of a device
- Using the Device Security Manager to change the security configuration of a device
- Provisioning the development certificates on a device
- Signing a file with the development certificates
- Provisioning a locked device
- Learning More About Mobile2Market

Scenario

This lab is performed using the device emulator that ships with Visual Studio 2005 to reduce complexity, save time, and protect your physical device from accidental provisioning. You may see references to a device, which simply refer to the emulator. You may also perform this lab on an actual Pocket PC or Smartphone device, provided it is not locked by the mobile operator or OEM.

Estimated Time to Complete This Lab

60 Minutes

Computer used in this Lab



MEDC

Exercise 1

Using the Device Security Manager to Determine the Security Configuration of a Device

Scenario

In this exercise, you will learn about the different security configurations on Windows Mobile–based devices, and you will also be able to determine the configuration on your device.

The Windows Mobile Application Security model implements a security-through-identity model that allows authenticated applications to execute on nearly all shipping devices. In other words, the device will trust a given application based on who signed the application (the issuer of the certificate that the application is signed with).

Windows Mobile–based devices have certificate stores that contain a set of certificates. There are three certificate stores on the device that define the trust level of your application: a privileged certificate store, a normal certificate store, and a Software Publisher Certificate (SPC) certificate store.

There are three levels of authentication: signed for privileged execution, signed for normal execution, and unsigned. The level of authentication that is required to run an application depends on the application programming interfaces (APIs) that the application uses.

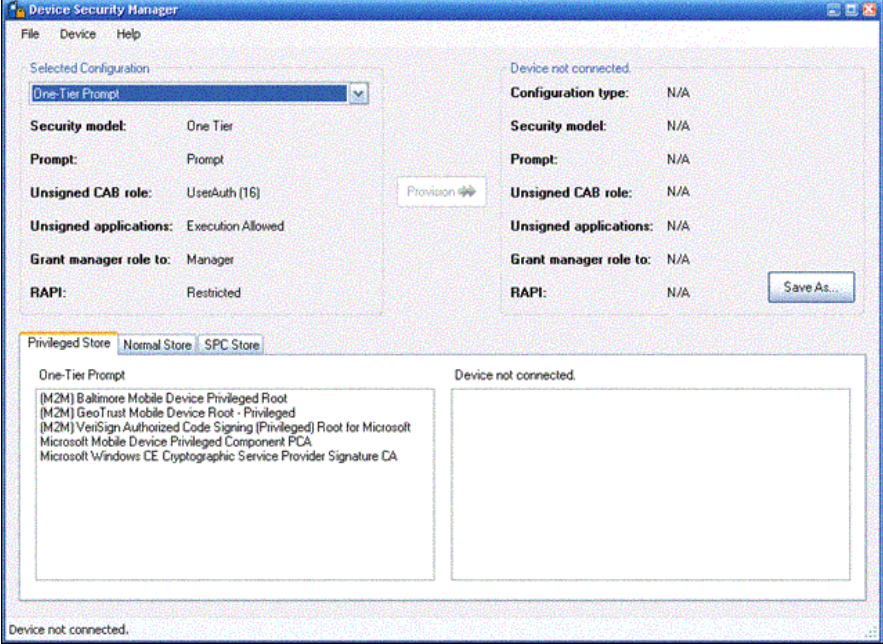
A *privileged application* is signed with a certificate that is in the privileged certificate store on the device. A normal application is signed with a certificate that is in the *normal* certificate store on the device. An application that is *unsigned* has no certificate.

Privileged certificate is a certificate that is in the privileged certificate store on a specific device. Note that there is nothing intrinsic to the certificate itself that is privileged. It is only the presence of the certificate in the privileged certificate store on a particular device that makes the certificate privileged.

The term *authentication* refers to how an application is signed. The execution permissions that an application receives are based on the security configuration on the device and the level of authentication of the application. The distinction between authentication and permission is important; for example, it is entirely possible to have an unsigned application run in privileged execution mode.

In this exercise, you will use the Device Security Manager tool to understand the most common security configurations.

Tasks	Detailed Steps
1. Open the Device Security Manager	a. On the desktop computer, click Start All Programs Device Security Manager Device Security Manager . The Device Security Manager application opens, and it should look like Figure 1.

Tasks	Detailed Steps
	 Figure 1. Device Security Manager <i>Note: Notice the two main groups of information. On the left, there is a Selected Configuration group that shows you the security policies of the selected configuration. On the right, there is a similar group that shows the security policies of the connected device. The tabs at the bottom let you see the certificates on the selected configuration and the connected-device configuration.</i>
2. Select a configuration	a. In the Selected Configuration box in the Device Security Manager, select the Security Off configuration. The security policies update to reflect the security policies for the selected configuration. <i>Note: The following are descriptions of each of the security policies:</i> Security Model The security model can be One Tier or Two Tier. In a two-tier security model, applications run with either privileged permissions or normal permissions. In a one-tier security model, applications always run in privileged mode. Both two-tier and one-tier models allow unsigned applications to be blocked from running. Prompt The prompt policy can be Prompt or No Prompt. If set to Prompt, the user will be prompted to decide if an unsigned .cab file should be allowed to deploy or not. Unsigned CAB Role and Grant Manager Role If the unsigned CAB role policy is among the grant manager policy roles, unsigned .cabs will run with a Manager role. The manager role usually has privileged mode execution permission. Unsigned Applications The unsigned applications policy determines whether unsigned applications are allowed to execute or not. RAPI The remote API (RAPI) policy can be Allowed (all RAPI calls are allowed), Disabled (no RAPI calls are allowed) or Restricted. For more information about RAPI restricted mode, see RAPI Restricted Mode Security at http://msdn.microsoft.com/library/default.asp?url=/library/en-us/mobilesdk5/html/wce51conRAPIRestrictedModeSecurity.asp.

Tasks	Detailed Steps
	b. Click the Normal Store tab. This step shows the Normal Certificate Store certificates. An application signed with any of these certificates is able to execute in Normal Execution mode. Some of the certificates start with the (M2M) prefix. These are Mobile2Market certificates on the device. More information about Mobile2Market is provided in Exercise 6. c. Click the SPC Store tab. This step shows the SPC Certificate Store certificates. A .cab or .cpf file signed with any of these certificates is able to install on the device with privileged execution mode. d. Click the Privileged Store tab. This step shows the Privileged Certificate Store certificates. An application signed with any of these certificates is able to execute on the device with privileged execution mode. e. Change the selected configuration to Locked and notice the certificates in privileged store are different that in the Security Off configuration. For example, the Locked configuration does not have Mobile2Market certificates in any of the configurations. <i>Note: If you look at the different configurations, you will notice the different security policies for each of the configurations. The following are descriptions of each of the standard configurations:</i> Locked or Mobile2Market Locked <i>With a Locked configuration, the only applications that will run are those applications that have been signed with a certificate in one of the device's certificate stores. Moreover, the use of the certificates that are in the certificate stores is controlled completely by the original equipment manufacturer (OEM), the mobile operator, or Mobile2Market.</i> Two-Tier-Prompt <i>In a Two-Tier Prompt device, the only applications that will run with privileged execution mode are those that are signed with a certificate in the privileged store. Applications that are signed with a certificate in the normal store will run with normal mode execution. Applications that are unsigned will only be allowed to run if you respond affirmatively to the security prompt that appears when you try to run a given unsigned application. After you have allowed a given application to run by responding affirmatively to the prompt, the application will always run in normal execution mode.</i> <i>Note: Now you have an idea about the different standard security configurations, and you should understand the different security policies.</i>

Exercise 2

Using the Device Security Manager to Change the Security Configuration of a Device

Scenario

In this exercise, you will learn how to determine a device's current security configuration and how to provision it with different security configurations. You may find it useful to provision your device with different security configurations, so you can understand how your application will run on each of the security configurations.

Tasks	Detailed Steps
1. Determine a device's security configuration	a. If Device Security Manager is not currently running, open it by clicking Start All Programs Device Security Manager Device Security Manager. <i>Note: You will now connect the device emulator to ActiveSync to change its configuration.</i> b. On the desktop computer, open Visual Studio 2005. c. In Visual Studio, click Tools Device Emulator Manager. d. In the Device Emulator Manager, right-click Windows Mobile 5.0 Smartphone Emulator, and then select Connect. The emulator opens. <i>Note: Wait until the emulator shows the Today screen.</i> e. In the Device Emulator Manager, right-click Windows Mobile 5.0 Smartphone Emulator, and then select Cradle. This step creates an ActiveSync connection with the emulator. f. Dismiss the ActiveSync message that appears by clicking OK. g. Dismiss the Synchronization Setup Wizard dialog box that appears by clicking Cancel. <i>Note: After you have established an ActiveSync connection, Device Security Manager detects your connected device. The Connected Device Configuration gets populated with data about your device. To understand what each of the security policies mean, see Exercise 1.</i> <i>Note: Because you connected to the Smartphone emulator, it has a two-tier security configuration. To change the security configuration by using the Device Security Manager, you will change its configuration to a One-Tier Prompt security configuration.</i>
2. Provision your device with a different security configuration	a. While your emulator is connected, select One-Tier Prompt as your selected configuration. b. In the middle of the Device Security Manager, click the Provision button. <i>Note: The first time Device Security Manager attempts to provision your emulator, it deploys a .cab file that needs to execute in privileged mode. However, the .cab file is not signed. Therefore, in some security configurations, you may need to respond affirmatively to a prompt on your emulator for the .cab file to execute in privileged mode.</i> <i>Note: Notice that the status bar message changes to inform you that the device is being provisioned with the selected configuration. If your device cannot be provisioned because it is either Locked, Mobile2Market Locked, or Two-Tier Prompt, you may not be able to provision your device.</i> c. If your device is properly provisioned, you will see a message that indicates that

Tasks	Detailed Steps
	the device is provisioned. Click OK to close the message. <i>Note: In the case of the Windows Mobile 5.0 Smartphone Emulator, you should also click Done on the device emulator.</i> <i>Note: Notice that the security policies change to match the configuration that you provisioned.</i>

Exercise 3

Provisioning the Development Certificates on a Device

Scenario

In this exercise, you will be introduced to the SDK certificates, why they are important, and how you can provision your device with them.

Note: Depending on the device's standard configuration, you need to prepare your device for development. The following descriptions will help you understand how to prepare your device:

Locked or Mobile2Market Locked

If you are using a physical device that has the Locked or Mobile2Market Locked configuration, the only applications that will run are those applications that have been signed with a certificate in one of the device's certificate stores. Moreover, the use of certificates that are in the certificate stores is controlled by the OEM, the mobile operator, or Mobile2Market. Because these certificates are private (that is, their private keys are secret), you cannot use them to sign your application during day-to-day development. Instead, you need to install other certificates in the certificate store, and then sign your application with one of them. Microsoft ships a set of certificates and corresponding private keys in the Windows Mobile 5.0 SDK for this purpose. You can find these certificates in the SDK Tools directory packaged in SdkCerts.cab.

One caveat is that only privileged processes can install certificates. Therefore, the OEM or mobile operator must offer a developer program that you can use to install these certificates or to allow you to change the configuration of the device. The OEM or mobile operator can do so in several ways; for example, a Web site that provides an executable file that installs the SDK certificates on your device. This executable file will likely be keyed with the unique identifying number of your device, so it will not work on any other device.

Two-Tier-Prompt

If your application needs to run in privileged mode, you need to install the SDK certificates exactly as you would on a Locked or Third-Party Signed device.

If your application does not need to run in privileged mode, you do not need to install the SDK certificates because you can always respond affirmatively to the security prompts. However, to avoid the inconvenience of being prompted, you can install the SDK certificates exactly as you would on a Locked or Third-Party Signed device.

Note that if you respond affirmatively to a prompt, you will not be prompted again for that module. But if you recompile, the recompiled module is considered a new module, and you will be prompted again.

One-Tier-Prompt

On a One-Tier-Prompt device, install the SDK certificates by running SdkCerts.cab on the device. You can find this file in the Tools directory of the SDK.

On Pocket PC, you can use ActiveSync to copy SdkCerts.cab to the device, and then open it in File Explorer.

On Smartphone, use ActiveSync to copy it to \Windows\Start Menu\Accessories, browse to Start Menu\Accessories on the device, and then open the file.

Security-Off Device

On a Security-Off device, you do not need to install SdkCerts.cab. Note, however, that it is not recommended to use the Security-Off configuration, and there will not be any retail devices that ship with this configuration.

Emulator

The SDK certificates are prebuilt into the emulator, so you do not need to install them yourself.

Even though the emulator already has the SDK certificates by default, they will be removed when you provision it with one of the standard configurations from Device Security Manager.

Tasks	Detailed Steps
1. Provision development	a. Be sure the Smartphone emulator is connected to ActiveSync and is detected by

Tasks	Detailed Steps
certificates by using the Device Security Manager	the Device Security Manager. <i>Note: If it is not connected, follow the steps in Exercise 2 to connect it.</i> b. Click the Privileged Store tab. c. If you provisioned the Smartphone emulator with the One-Tier-Prompt configuration as described in Exercise 2, you will notice that there is no certificate named “TEST USE ONLY – Sample Privileged Root for Windows Mobile SDK.” d. On the Device Menu, click Add Development Certificates. <i>Note: Notice the status bar is updated to describe that the device is being provisioned. Wait for the SDK certificates to be added to your emulator.</i> e. If provisioned successfully, you will see a message that indicates that the device was provisioned successfully. Click OK to dismiss the message. <i>Note: Notice that some certificates were added to the certificate stores. For example, in the Privileged Store of your connected device, you should be able to see the “TEST USE ONLY – Sample Privileged Root for Windows Mobile SDK” certificate.</i> <i>Note: You will now remove SDK certificates from the emulator to add them by using SdkCerts.cab. You will remove the SDK certificates by provisioning one of the standard configurations on top of the current configuration.</i>
2. Provision with one of the standard configurations	a. Follow the steps in Task 2 “Provision your device with a different security configuration” of Exercise 2. Provision your device with the One-Tier-Prompt configuration. b. Verify that you do not see any certificates that start with TEST USE ONLY in the certificate stores of the connected emulator.
3. Provision development certificates by using SdkCerts.Cab	a. Be sure the Smartphone emulator is connected to ActiveSync and is detected by the Device Security Manager. <i>Note: If it is not connected, follow the steps in Exercise 2 to connect it.</i> b. Open Windows Explorer and browse to the tools directory of the Windows Mobile 5.0 SDK (that is, C:\Program Files\Windows CE Tools\wce500\Windows Mobile 5.0 Pocket PC SDK\Tools). c. Right-click the SdkCerts.cab file, and then click Copy. d. Close Windows Explorer. e. In Microsoft ActiveSync, click Explore. f. In the Mobile Device Explorer window, double-click My Windows Mobile-Based Device, and then navigate to Windows\Start Menu\Accessories. g. Right-click in an empty part of the explorer window, and then click Paste. h. On the File Conversion message, click OK. i. After the file is copied, close the device’s Accessories window. j. On the Smartphone emulator, click the left soft key (under the Start menu). k. Click the soft key again (under More), and then use the direction keys to select Accessories, and then click the Action key, which is in the center of the navigational pad. l. Use the direction keys to move to SdkCerts.cab, and then click the Action key to open the file. m. Be sure to respond affirmatively to the messages on the device. n. After the installation is complete, click the soft key under Done. o. In Device Security Manager, press F5 to refresh the connected device configuration. <i>Note: It may take up to ten seconds for the screen to refresh with the new information.</i>

Tasks	Detailed Steps
	<i>Note: Notice that some certificates were added to the certificate stores. For example, in the Privileged Store of your connected device, you should be able to see the TEST USE ONLY – Sample Privileged Root for Windows Mobile SDK certificate.</i>

Exercise 4

Signing a File with the Development Certificates

Scenario

In this exercise, you will learn how to sign a file with the development certificates by using several different tools.

Tasks	Detailed Steps
1. Add the development certificates to your desktop computer's certificate store	- Click Start Run. - Type C:\Program Files\Windows CE Tools\wce500\Windows Mobile 5.0 Pocket PC SDK\Tools\sdksamplePrivDeveloper.pfx, and then click OK. The Certificate Import Wizard should appear. - Click Next. - Click Next (the path to the certificate should be filled in for the File name). - Click Next (you do not need a password). - Click Next (leave the selected default options as they are). - Click Finish. - In the Security Warning message, click Yes. - In the Certificate Import Wizard success message, click OK. - Repeat Steps a to i for SDKSampleUnPrivDeveloper.pfx within the same path.
2. Sign an application by using the Device Security Manager	- Start Device Security Manager if it is not already running. - Click File Sign File. - You can choose whether to sign with the Normal or Privileged Development certificate. For this step, choose Normal. - Select a file to sign. You can sign the ToSign.exe file, which is located at C:\Program Files\Windows Mobile Developer Samples\HOLs\HOL313_WM_Device_Security\Unsigned. Click Open and wait for the file to be signed.
3. Check the certificate of a file	- On the desktop computer, open Windows Explorer. - Browse to the file of interest. In this case, you can verify that the ToSign.exe file was properly signed. Browse to C:\Program Files\Windows Mobile Developer Samples\HOLs\HOL313_WM_Device_Security\Unsigned. - Right-click ToSign.exe, and then click Properties. The File Properties dialog box appears. - Click the Digital Signatures tab. You should see the certificate that the file is signed with, as shown in Figure 2.

Tasks	Detailed Steps
	Figure 2. The certificate of a file in the File Properties dialog box e. Click Cancel to close the Properties window. f. You can also check the signature on a file in the Device Security Manager by clicking File Check File Signature and then browsing for the file. <i>Note: Visual Studio allows you to sign your projects as part of the build process.</i>
4. Sign a native application from a Visual Studio 2005 project	a. Start Visual Studio 2005 if it is not already running. b. Click File Open Project/Solution. c. Open the solution called HelloWorld.sln, which is located at C:\Program Files\Windows Mobile Developer Samples\HOLs\HOL313_WM_Device_Security\HelloWorld. This solution contains two projects. A managed project called ManagedHelloWorld and a native project called NativeHelloWorld. d. In Solution Explorer, right-click the NativeHelloWorld project. e. Click Properties. f. Under Configuration Properties, select Authenticode Signing. A form should appear similar to the one in Figure 3. Figure 3. The Authenticode Signing form g. In the Authenticode Signature box, select Yes. h. In the Certificate box, click the ellipsis button. A dialog box appears that lets you select from the certificates installed on your desktop computer for which you have a private key. Because you have installed the SDK certificates, you should be able to pick from a set of certificates that start with TEST USE ONLY.

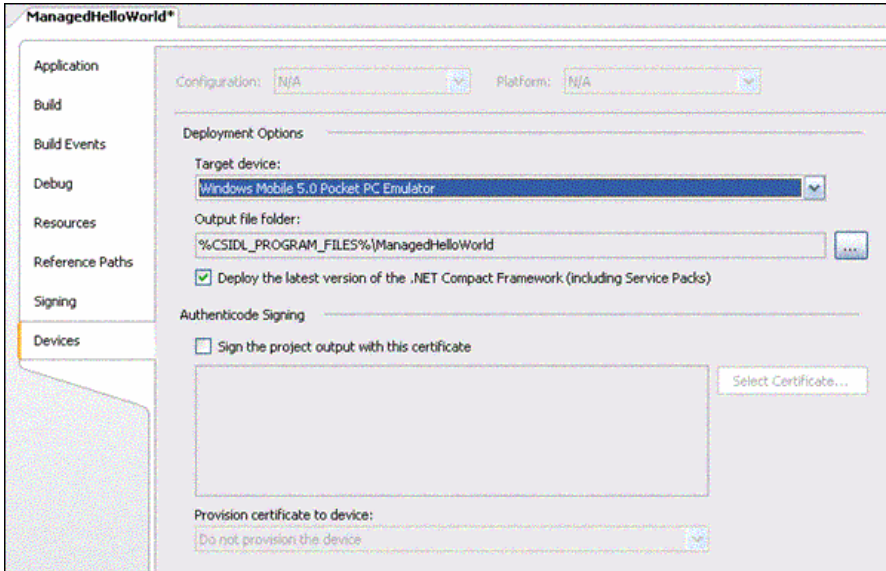
Tasks	Detailed Steps
	i. Select the unprivileged SDK certificate, which has the issuer name of TEST USE ONLY – Sample Unprivileged Root for Windows Mobile SDK. j. Click OK to confirm the selection. k. On the project's Property Pages dialog box, click OK. l. In Visual Studio, click Build Build Solution. This step produces the NativeHelloWorld.exe file, and it will be signed with the privileged development certificate. m. Verify that the file has been signed by doing the "Check the certificate of a file" task. <i>Note: The path to the file is C:\Program Files\Windows Mobile Developer Samples\HOLs\HOL313_WM_Device_Security\HelloWorld\NativeHelloWorld\Windows Mobile 5.0 Smartphone SDK (ARMV4I)\Debug\NativeHelloWorld.exe.</i>
5. Sign a managed application from a Visual Studio 2005 project	a. In Solution Explorer, in Visual Studio 2005, right-click the ManagedHelloWorld project. b. Click Properties. c. Click the Devices tab to Authenticode sign your application, as shown in Figure 4. <i>Note: Do not click the Signing tab. The Signing tab refers to strong naming (specific to managed code), but the Devices tab allows you to Authenticode sign.</i>  d. Select Sign the project output with this certificate. e. Click the Select Certificate button. The certificate selection dialog box appears. f. Select the unprivileged certificate (the issuer is TEST USE ONLY – Sample Unprivileged Root for Windows Mobile SDK). g. Click OK to confirm the selection. h. Close the project's Properties dialog box. i. In Visual Studio, click Build Build Solution. This step builds DeviceApplication1.exe, and it will be signed with the privileged development certificate. j. Verify that the file has been signed by doing the "Check the certificate of a file" task.

Figure 4. The Devices tab

Tasks	Detailed Steps
	Note: The path to the file is C:\Program Files\Windows Mobile Developer Samples\HOLs\HOL313_WM_Device_Security\HelloWorld\ManagedHelloWorld\bin\Debug\DeviceApplication1.exe. Note: Because all you are doing in the Hello World application is showing a message, we needed to sign it with the normal SDK certificate. If you had used an API that required privileged access, you should have signed with it with the privileged SDK certificate. For more information about which APIs require privileged mode execution, see Trusted APIs at http://msdn.microsoft.com/library/default.asp?url=/library/en-us/wceosdev5/html/wce50conTrustedAPIs.asp.

Exercise 5

Provisioning a Locked Device

Scenario

There are some situations when your device may be Mobile2Market Locked or Locked, and it may not have the SDK certificates for development. In this exercise, you will lock the Smartphone emulator to see what would happen if you tried to change the configuration of a locked device by using the Device Security Manager.

Tasks	Detailed Steps
1. Lock your emulator and try to provision it	a. Be sure the Smartphone emulator is connected to ActiveSync and is detected by the Device Security Manager. <i>Note: If it is not connected, follow the steps in Exercise 2 to connect it.</i> b. Provision your emulator with the Locked configuration by following the “Provision your device with a different security configuration” task in Exercise 2. <i>Note: Next, you will remove the Configuration Manager Tool certificate from the emulator. This certificate is not found on locked devices.</i> c. Open Windows Explorer and browse to C:\Program Files\Windows Mobile Developer Samples\HOLs\HOL313_WM_Device_Security. d. Use ActiveSync to copy RemoveToolCert.cab to the \Windows\Start Menu\Accessories folder on your emulator. e. On the emulator browse to Start Menu\Accessories, and then open the RemoveToolCert file. <i>Note: For more information about copying a file to the device emulator and opening a file on the emulator, see the "Provision development certificates by using SdkCerts.Cab" task in Exercise 3.</i> f. Be sure to respond affirmatively to any prompts on your device. g. From Device Security Manager, press F5 to refresh the connected device configuration. The Configuration Manager Tool certificate should not be in any of the certificate stores. h. Click Provision. A dialog box that explains that it is not possible for you to provision your device using Device Security Manager should appear. At this point, you know that you should contact your mobile operator to inquire about their development program. i. Close the message.

Exercise 6

Learning More about Mobile2Market

Scenario

Mobile2Market is a program that Microsoft operates for independent software vendors and that provides a unified market for mobile applications. The Mobile2Market code signing program is open to all Windows Mobile application vendors—large or small. It enables you to sign your application with the Mobile2Market certificates that are shipped on almost 100 percent of the Windows Mobile devices that are sold. Using Mobile2Market's certificate ensures that your application will be able to meet the security requirements of almost every device on the market today. Signing up to use Mobile2Market is a simple process.

Tasks	Detailed Steps
1. Sign up to use Mobile2Market	Note: Establish an account with a vendor that supports Mobile2Market. Currently there are two choices: Verisign and GeoTrust. Both vendors offer reliable, competitively priced packages to get your applications signed. For more information about the current pricing plans, see the following Web sites: Note: Verisign at http://www.verisign.com/products-services/security-services/code-signing/microsoft-smartphone-code-signing. Note: GeoTrust at http://www.geotrust.com/products/signing_services/code_signing_windows_mobile.asp. These vendors are also known as Certifying Authorities (CA). To establish an account with a CA, a brief background check will be conducted to confirm your identity. Usually, this background check includes verifying your phone number, company address, and Web site. After your account has been established, the CA issues you a USB key that allows you to sign executable files with a Mobile2Market public key. You upload signed files to an online tool on the CA's Web site, which gives your files a corresponding Mobile2Market private key. At this point, your signed files are ready for distribution. Each time you sign a file, a signing token is used. There is a cost associated with each token, so it is best to leave Mobile2Market signing until the end of your product development cycle.

Summary

In this lab, you performed the following exercises:

- Using the Device Security Manager to determine the security configuration of a device
- Using the Device Security Manager to change the security configuration of a device
- Provisioning the development certificates on a device
- Signing a file with the development certificates
- Provisioning a locked device
- Learning more about Mobile2Market

After you complete this lab, you should have a good idea about the Windows Mobile security model and should be know the following: how to determine what authentication level your applications should have, how to use the many different tools that are available to you to test your application's execution on different security configurations, and how to use the tools that are available to you to sign your applications during development.