Implementing Forefront Identity Manager 2010

Student Manual

Module 8: Other Considerations

Table of Contents

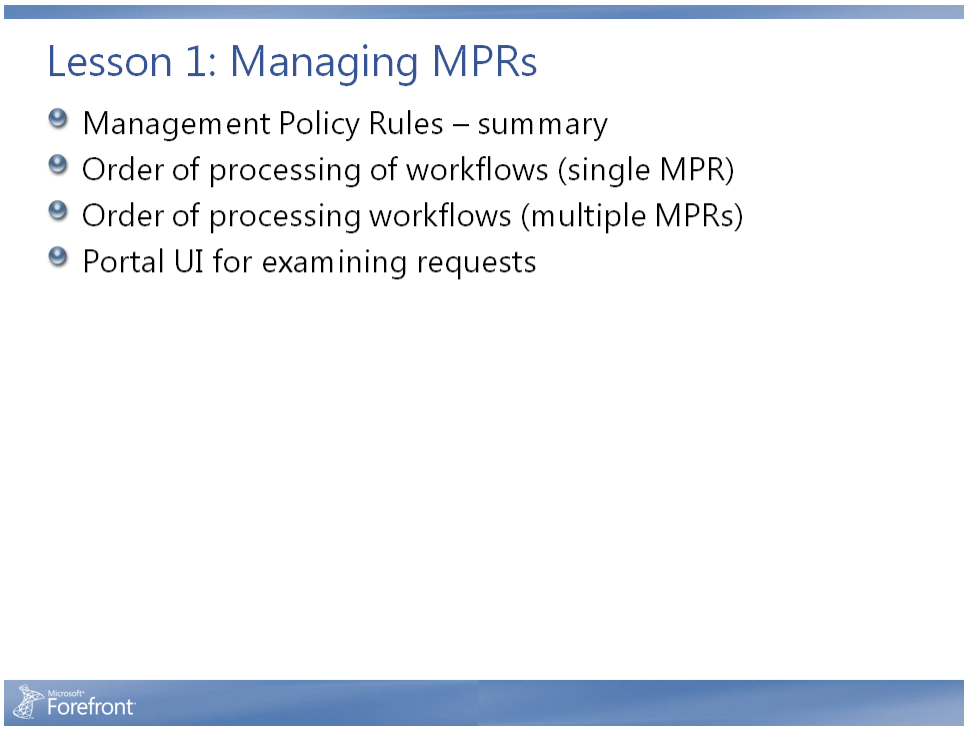
Module 8: Other Considerations

Module Overview

During the course we have made use of Management Policy Rules (MPRs) without much of a formal introduction. This module does that, and then goes on to discuss additional features of MPRs, including the User Interface (UI) that is used to troubleshoot them.

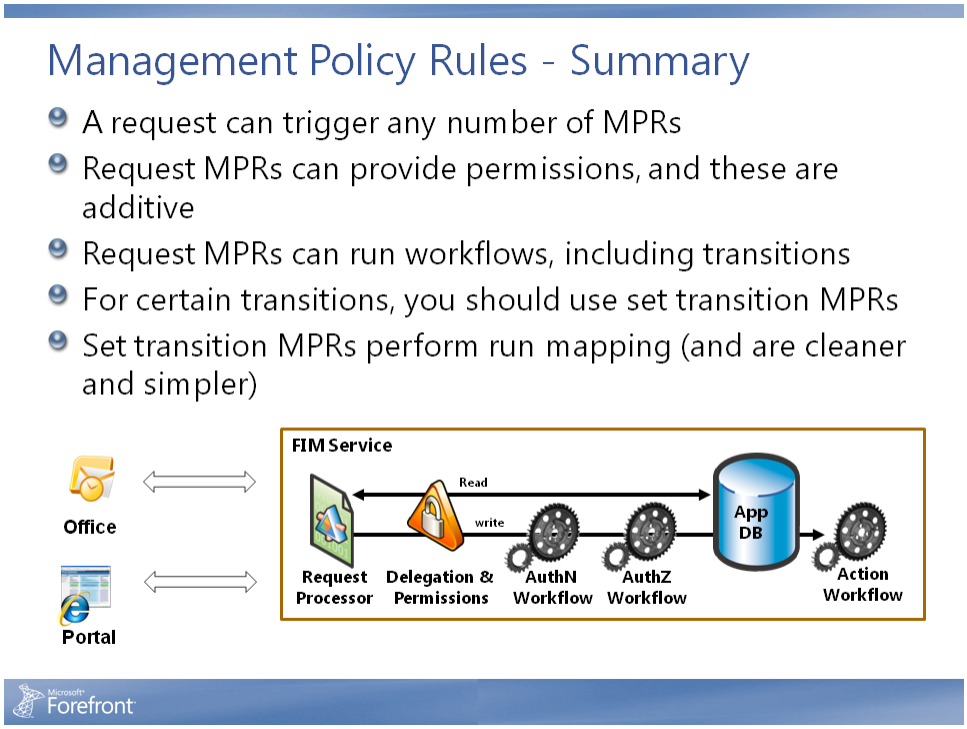
The second part of the module focuses on some operational considerations. A course of this level can't cover every topic to do with operating Microsoft® Forefront™ Identity Manager 2010 (FIM), but we start by covering such topics as backup and restore, and scripting run cycles.

Lesson 1: Managing MPRs



MPRs are perhaps the most fundamental building block of the FIM service. They control permissions to the portal, and run the workflows that make things happen.

Management Policy Rules – Summary



MPRs

We use MPRs to moderate and control the requests to perform operations on portal data. A request can trigger any number of MPRs, and those MPRs can grant permissions and run workflows.

There are two types of MPR: request and set transition.

Permissions

Before anything can happen, permission has to be established. Request MPRs can grant permissions. Permission-granting MPRs grant permission for a requestor set to perform operations on particular attributes of target sets (potentially with the added restriction of a transition).

> **Note**: In order to be recognized at all, the portal must have the objectSID for a user, so you would flow this from Active Directory® (AD). Also note that the forms used to edit resources, Resource Control Display Configurations (RCDCs), may further restrict you. They could offer read-only access to an attribute, even though a user might have modify permission.

Additive Permissions

MPR permissions are additive. For example, one MPR might give read permission to any requestor for all attributes of the target object, and another might give modify permission to some attributes provided the requestor is in the All HR set, while yet a third gives modify permission to some other attributes provided the requestor is in the All Managers set. So the net effect is that the requestor (an HR manager) may have both read and modify access to the various attributes mentioned.

Permitting Transitions (or Not)

Permissions can be made dependent on the target object being a member of a set (and may be persisting as a member of that set even after the operation). So an MPR can control permissions to make modifications that cause particular state changes, or transitions. It could, for example, detect a request to change a target such that the target will:

- Start off in a general set (for example, All People) and as a result of the modification fall into a particular set (for example, All Full Time Employees), whether or not it is still in the more general set.
- Start off in a particular set (for example, All Contractors) and as a result of the modification no longer be a member of that set, but still stay within the more general set (for example, All People).
- Transition from one particular set to another particular set (for example, from All Full Time Employees to All Contractors).

The first and last of these are not the same because there are other states, like being in All Interns. An MPR could grant permissions for someone to change a target so that it changes from All Full Time Employees to All Contractors, and yet not grant permission when they attempt to alter the target so that it changes from All Contractors to All Full Time Employees.

An example that has already arisen is that group scopes need regulating in exactly this way. You can change scope from universal to global, but not from domain local to global. Active Directory Users and Computers (ADUC) enforces this, and we can enforce it in FIM too, using permission-granting MPRs and sets.

Running Workflows

For read operations, that's it—we establish permission and perform the read. For write actions, there may be workflows to run. We are still talking here about request MPRs.

The authorization and authentication workflows look like gates, whereas action workflows take effect after the request has achieved its goal (after the database has been written to). Workflows can be very long in duration (think of an approval which could take days or weeks), so since even action workflows could take a long time, the request may not technically complete for a long time after the data is written.

Simple Cases

In the simplest case, we characterize an MPR behavior as "If this resource wants to do that operation to that resource, we must first run these authentication workflows, then (if that was okay) run these Authorization Workflows, and then (if that was okay) perform the operation, and finally run these Action Workflows."

Transitions

Where we are dealing with transitions, things get more complicated. If a request wants a resource to transition from a before set to an after set, we can permit that with an MPR, and (for example) also ask for additional authorization (perhaps asking a manager to approve it) and/or run an action workflow, such as a notification.

However, for some actions, particularly where we are provisioning and deprovisioning, we should use set transition workflows because:

- They are simpler to configure.
- They are simpler for someone else to subsequently comprehend.
- It is cleaner to separate permissions from these kinds of actions.
- They can make use of Run on Policy Update.

Set Transition MPRs

These are simply configured to run workflows on resources when they transition into a set, or alternatively, when resources transition out of a set. The prime example is:

- Add an AD account for a user when they fall into the All AD Users set (needs an add workflow and a transition in MPR).
- Remove the AD account when they fall out of the set (needs a remove workflow and a transition out MPR. Note that the synchronization rule that is involved is configured to **Disconnect FIM resource from external system resource when Sync Rule is removed**).

Run Mapping

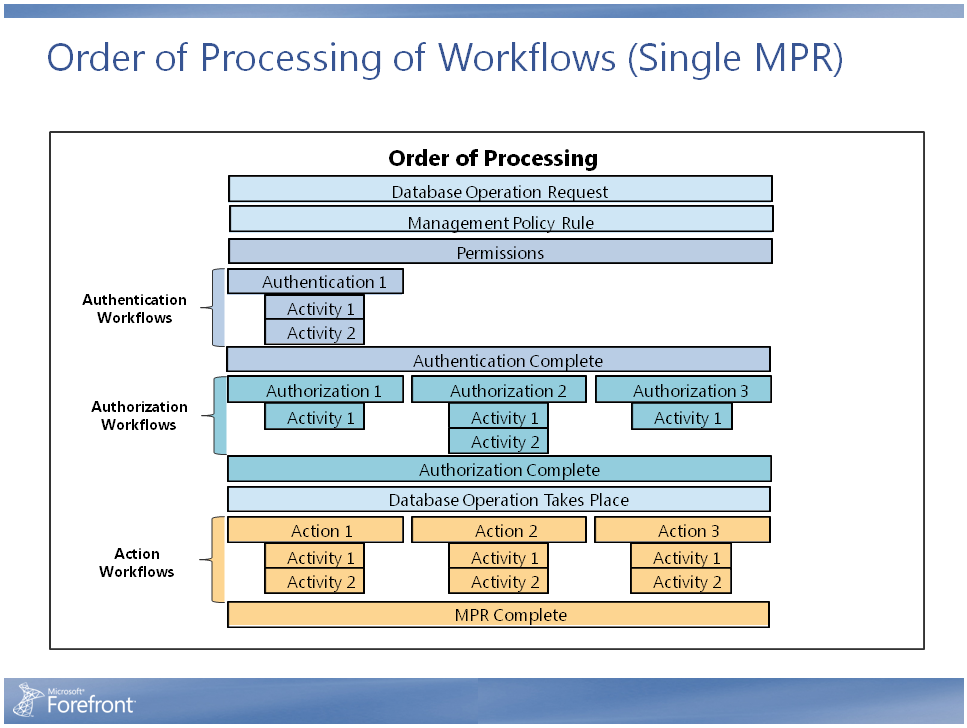
It is really important to consider what happens when you create or modify such MPRs, workflows, and sets, and you already have users, possibly in unknown states. The quick answer is to make sure that you mark the workflows for Run on Policy Update, so that they apply to existing resources. The longer answer is that it is a little more complicated and best explained by using some examples (which are based on some simple experimentation):

Set	Workflow	MPR	Run Mapping?
Create Set A	Create Workflow 1 (with Run on Policy Update)	Create Set Transition (In) MPR to run Workflow 1 on transition in to Set A	Workflow 1 is applied to all existing members of Set A
Edit user so they join Set A			Workflow 1 is applied to that user
	Edit Workflow 1 to do something else	Change MPR description	Nothing happens
	Change Workflow 1 again and Create Workflow 2 (with Run on Policy Update)	Add Workflow 2 to MPR	Workflow 2 is applied to all existing members of Set A

Set	Workflow	MPR	Run Mapping?
Edit Set A so that users fall into the Set			Both workflows are applied to the **new** members of Set A
	Create Workflow 3 (**with or without** Run on Policy Update)	Create Set Transition (Out) MPR to run Workflow 3 on transition out of Set A	Nothing happens
Edit user so they leave Set A			Workflow 3 is applied to that person
Edit Set A so that users drop out of the Set			Workflow 3 is applied to all users that dropped out

Note: Run on Policy Update only has an effect when a workflow is added to a transition in MPR (unless the MPR is disabled, in which case it fires when the MPR is enabled). You can sometimes achieve the desired update by removing the workflow from the MPR and adding it again. Also you can get a workflow to apply to non-members of a set by creating the anti-set and using a temporary transition in MPR.

Order of Processing of Workflows (Single MPR)



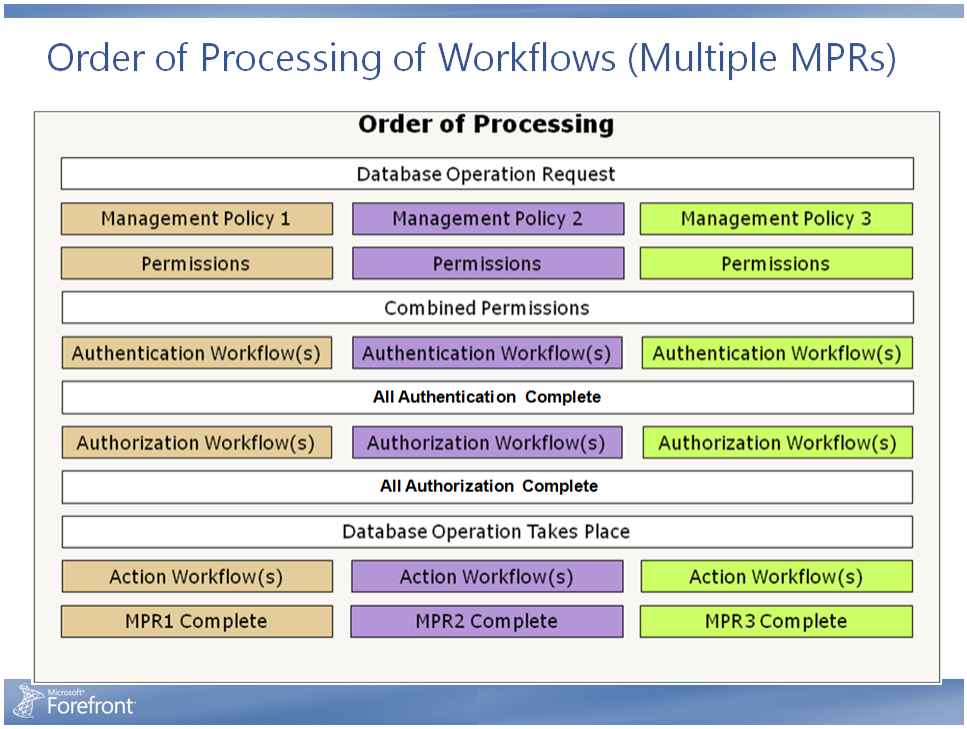
When an MPR runs, it can first grant permissions, and then it can run workflows. It can run a number of workflows for authentication, authorization, action, or none.

Workflows consist of activities. Each activity is itself a Windows® Workflow Foundation (WF), and that can be as complicated as a programmer can make it (for example, with branches and loops).

All authentication steps must be completed before any authorization steps are completed. Within authorization (for example), each authorization workflow runs in parallel while its individual activities run in series.

This is less complicated than it sounds!

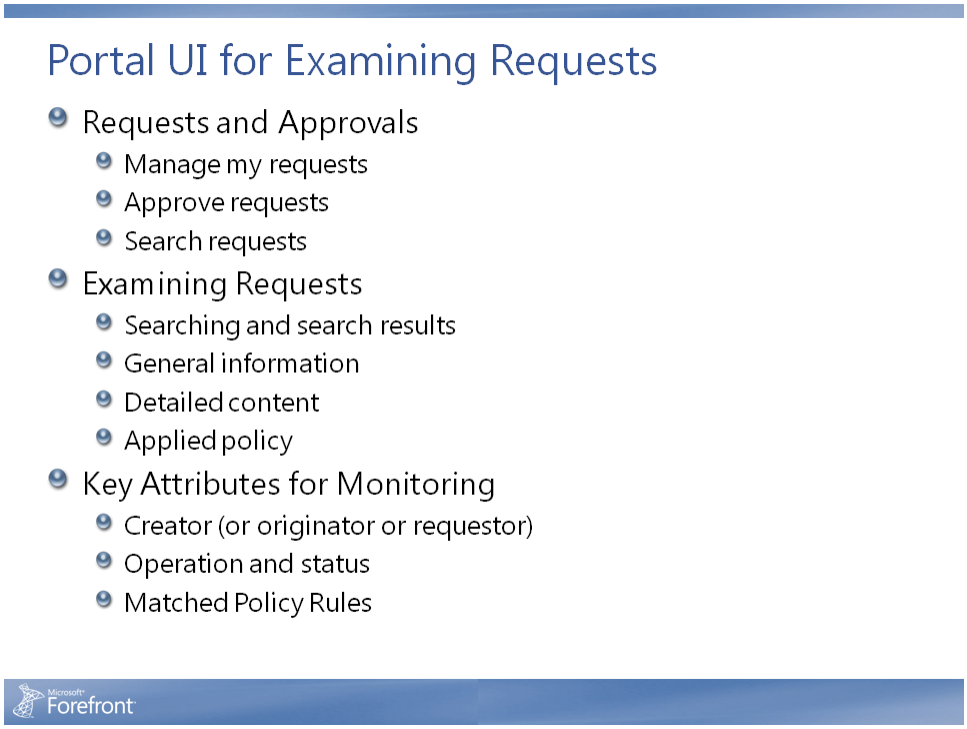
Order of Processing of Workflows (Multiple MPRs)



Any number of MPRs might be triggered by an attempt to update portal data, and each MPR can run authentication, authorization, and/or action workflows.

When several MPRs are triggered, these are processed in parallel, except that they must be tied together at various points so that permissions are combined, and authentication and authorization requirements are handled correctly.

Portal UI for Examining Requests



The portal logs requests in a good deal of detail so that troubleshooting and auditing can be performed. The status and content of the request can be examined, as well as the policy rules which were involved in the request.

Requests and Approvals

There is a parent navigation bar link for this, but it does the same as approve requests (below):

Manage My Requests

This allows you to list and examine requests you have made for the logged on user only. You will get one request for creating a group or user, another for updating a resource, and yet another for a deletion. You will generally see lots of requests that are completed, and maybe some that are pending approval (presumably by someone else), and maybe some that resulted in other states (like denied or authorizing).

Approve Requests

From here, you can list pending or completed requests that require your approval or rejection (for example, if you own a group someone wants to join). This is simply an alternative interface to the approvals sent in e-mail messages that you have already seen. Again, this is for the logged on user only.

Search Requests

This is more of an administrative option, allowing an administrator to look through all requests.

Examining Requests

In the search results, you can see the title of the request, the date it was created, the current status, the originator, and the operation (for example, create, delete, or modify (that is, update)).

The title of a request contains the action (for example, create, delete, or update), the object type (for example, person, group, synchronization rule), and the name of the affected object. Therefore, by careful use of even simple search criteria, you can locate the requests concerning any of these. For example, you could do a simple search on update or group or Max.

Using the advanced search you can specify any filter criteria you like, for example, request status Is denied. If you want those made by a particular user (which is variously displayed as the originator and the requestor, but is, in fact, the creator), you could use, for example, **Creator Is Max Benson**.

Once you have located the request of interest and opened it, the general information concerning a request is displayed, where you can see the approvals (if any) involved in the current request, as well as who took action (and what action) in an approval.

The detailed content page shows the details of the change that was requested.

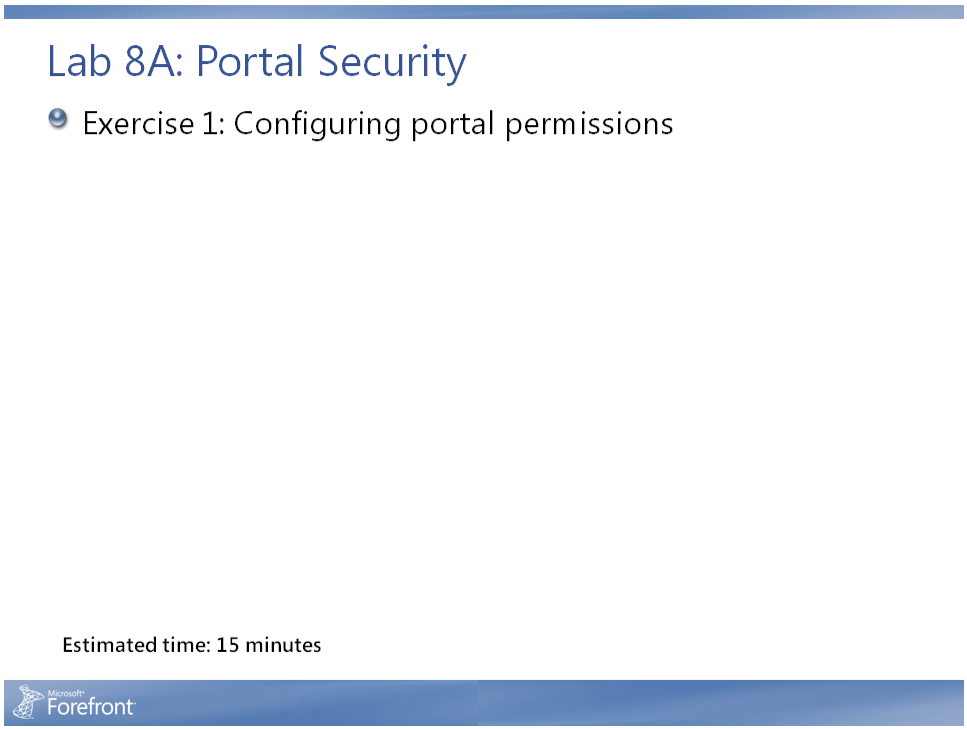
Applied policy shows the MPRs that were used to gain permission, or to execute this request. This is particularly useful when troubleshooting, since you can check that the MPRs that you expected to be involved actually were involved. If one is missing from this list, you have a problem with the MPR. Provided an MPR is displayed here, you can click through to the MPR, allowing you to drill down into the details of its configuration.

Key Elements for Monitoring

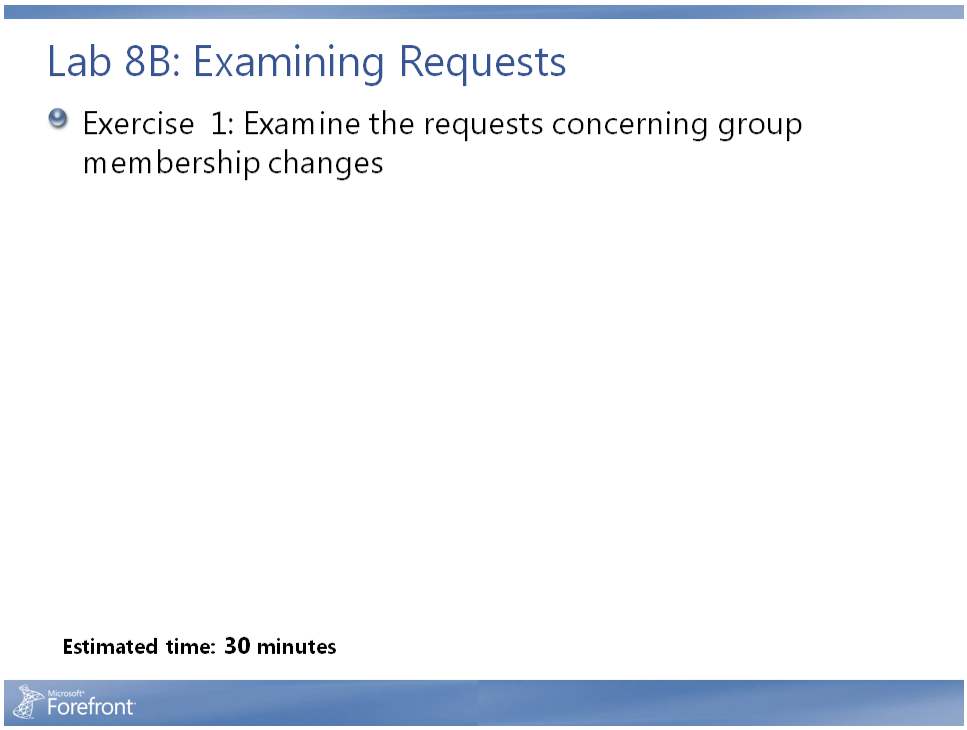
In summary, the key elements which you are likely to want to check during troubleshooting are:

- Originator – Used to locate the request.
- Operation – What change was requested?
- Status – Is the request authenticating, authorizing, validating, denied, or completed?
- Matched Policy Rules – Are all the MPRs in the list that are expected? Are those in the list correctly configured?

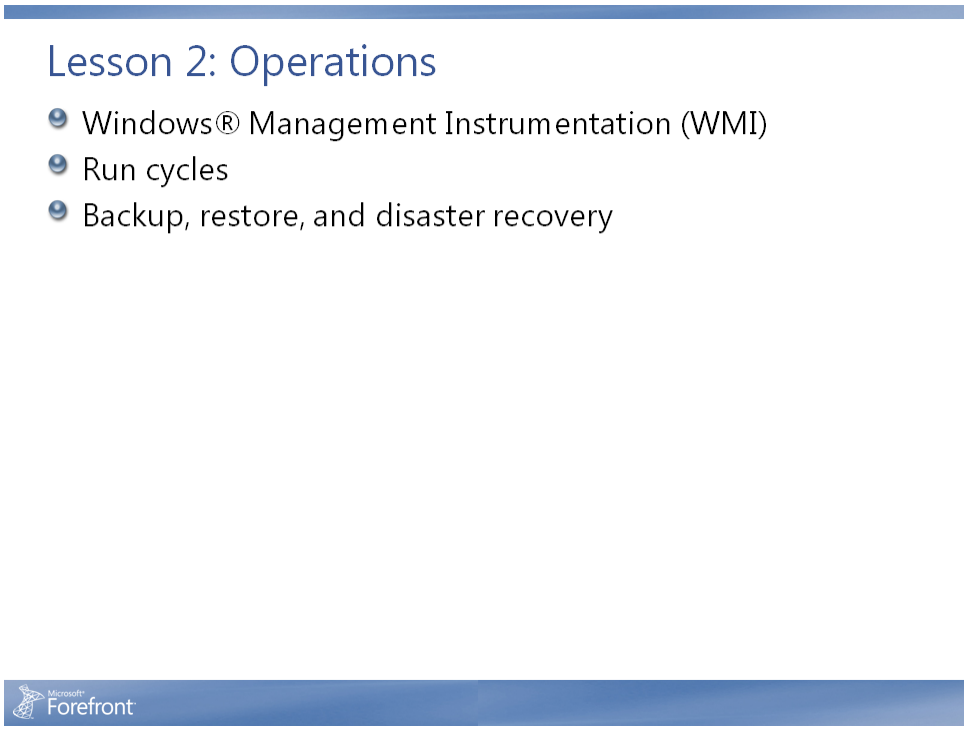
Lab 8A: Portal Security



Lab 8B: Examining Requests

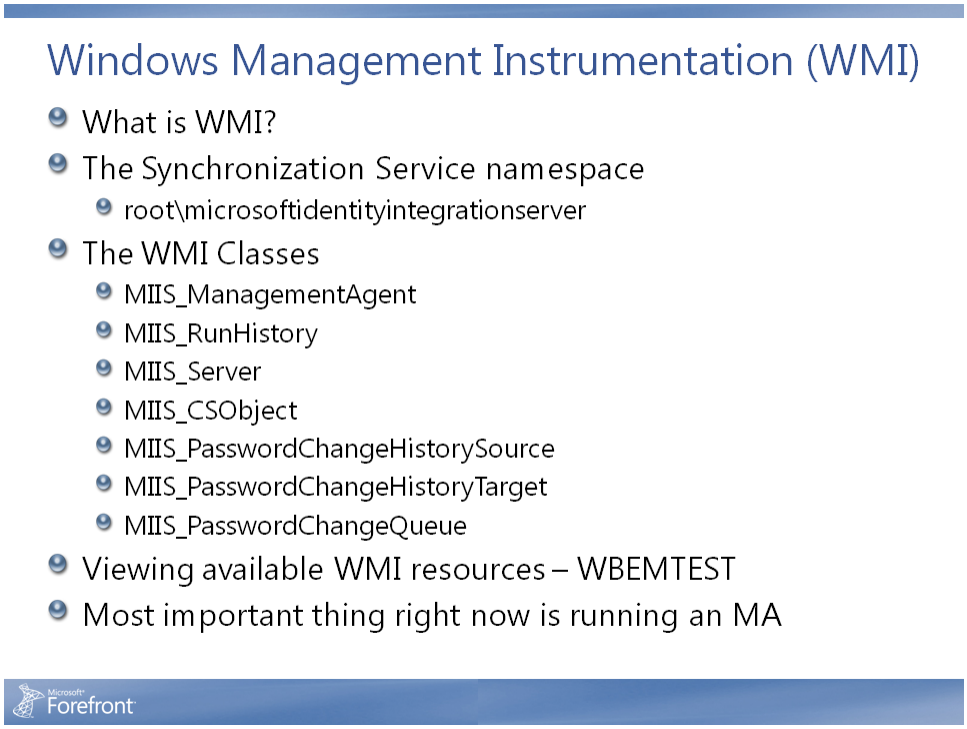


Lesson 2: Operations



This is a big topic, and could be more fully covered in a more advanced course. Here we cover organizing a cycle of MA runs, and backup/restore, enough for you to make a start.

Windows Management Instrumentation (WMI)



The Windows Management Instrumentation (WMI) interface can be used to control the synchronization service. In this course we are going to introduce it and use it just for running MAs, but we will at least mention what else can be done.

It is worth mentioning that FIM includes a Web service for communicating with the FIM service. So the portal uses the Web service to talk to the FIM service, as does the synchronization service manager (the FIM Management Agent (MA)) and the password reset tool. We could write our own entire portal if we so felt like it. Also, workflow activities use the Web service to communicate with the FIM service using provided WF tools. All of this is for a more advanced course!

What is WMI?

WMI is a component of the Microsoft® Windows® operating system and is the Microsoft implementation of Web-based Enterprise Management (WBEM), which is an industry initiative to develop a standard technology for accessing management information in an enterprise environment. WMI uses the Common Information Model (CIM) industry standard to represent systems, applications, networks, devices, and other managed components. You can use WMI to automate administrative tasks in an enterprise environment.

The purpose of WMI is to provide a standardized means for managing your computer system, be it a local computer or all the computers in an enterprise. In its simplest terms, management is little more than collecting data about the state of a managed object on the computer system and altering the state of the managed object by changing the data stored about the object. A managed object can be a hardware entity, such as a memory array, port, or disk drive. It can also be a software entity, such as a service, user account, or, in this case, an MA or the FIM synchronization service.

The Synchronization Service has its Own Namespace

The synchronization service has its own namespace within WMI (not renamed since a few versions ago):

```
root\microsoftidentityintegrationserver
```

When you are using any WMI tools or scripts, you will first need to connect to the WMI services using this namespace, and then you will have access to the various classes within WMI.

The WMI Classes

The classes of objects accessible through WMI are (not renamed from a few versions ago, yet):

Class	Purpose
MIIS_ManagementAgent	Provides method to Execute and Stop, and to return various statistics
MIIS_RunHistory	Information about historical executions of MAs
MIIS_Server	Access to the server name and clearing the stored MA run history
MIIS_CSObject	Access to Connector Space holograms; object GUIDs; this is used for setting and changing passwords
MIIS_PasswordChangeHistorySource MIIS_PasswordChangeHistoryTarget MIIS_PasswordChangeQueue	Three classes giving access to information about password changes requested, attempted, and pending

Viewing Available WMI Resources - WBEMTEST

The WMI Tester tool (wbemtest.exe, contained in the system32\wbem folder) allows you to access the WMI interfaces for exploration purposes.

Using WBEMTEST

To use this, locate wbemtest.exe in the system32\wbem folder and run it. Before you can execute commands, you must connect to the FIM (MIIS) WMI Provider. Click **Connect…** and provide the following namespace:

```
root\microsoftidentityintegrationserver
```

You will find that many of the buttons are not required for what we want to do. Usually you can simply double-click an existing result to drill down further.

Warning! You will see a Delete button on many dialog boxes, and often this really does delete a WMI class permanently. Don't click it.

Enumerating the WMI Classes Available

Click the **Enum Classes** button, and leave the superclass box empty. Simply click **OK**. Five system classes (prefixed with __), and the seven FIM (MIIS) classes appear.

Viewing the Methods and Properties for an Object Class

Having identified the object classes available, you can view the methods and properties available for a particular class. For example, you can double-click **MIIS_ManagementAgent**, and be presented with the object editor in which you can see the properties and methods associated with the MA class.

The interesting properties are easier to see if you select the **Hide System Properties** check box.

Viewing Instances of an Object Class

Once you have opened a class, you can list the instances of that class (in this case, the specific MAs) by clicking the **Instances** button.

Viewing Properties of an Object Instance

Once you have a list of MAs, you can access the properties of a particular MA by double-clicking its entry in the list. (You can also open an instance directly by clicking **Open Instance…** in the main Windows Management Instrumentation Tester (wbemtest) window, and supplying an object filter like **MIIS_managementagent.name="Fabrikam Telephone Txt MA"**).

You now see the properties for the particular MA.

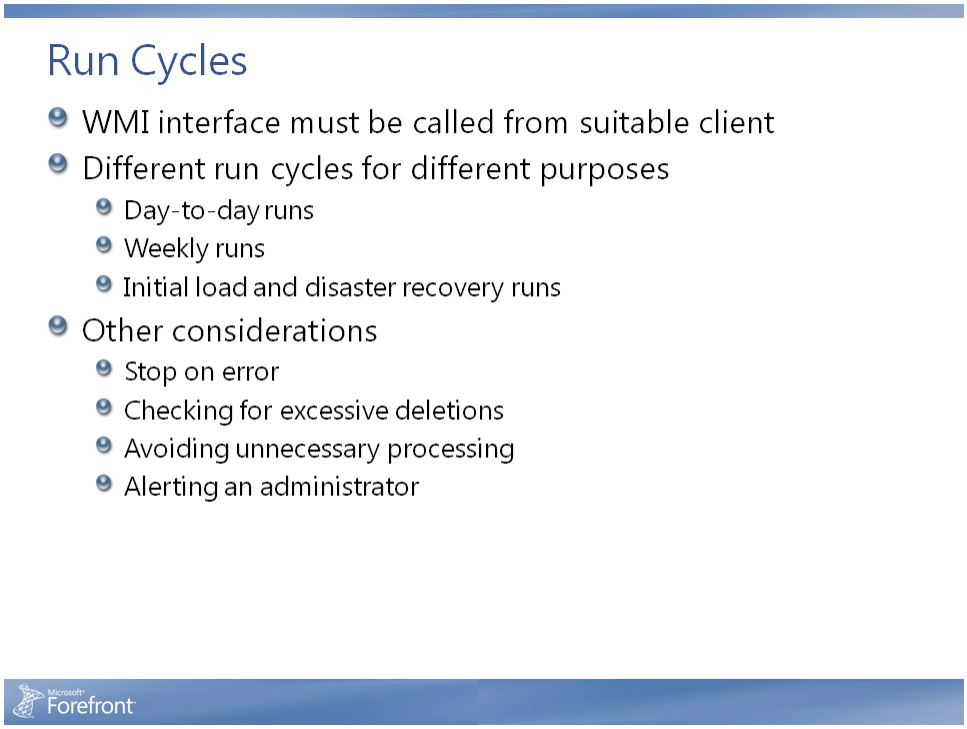
Executing WMI Queries

Using the Query… button, you can execute WMI Query Language (WQL) queries. Try this one:

```
select * from MIIS_managementagent
```

You should get a list of management agents. These SQL-like queries are also usable in scripts. Note that the MIIS_CSObject class only supports queries that contain a WHERE clause—you cannot request all CS objects at once.

Run Cycles



You will need to find a way to execute run profiles via the WMI interface, either using a script or program of your own, or perhaps using Windows® PowerShell™ commands, or by using a utility like the one available with the Microsoft® Identity Integration Server (MIIS) resource kit.

Run Cycles for Different Purposes

Remember that you may need different run cycles for different purposes.

Day-to-Day Running

Regular operation should almost always use delta synchronizations, and wherever possible, they should use delta imports. Exports are always deltas anyway. Full synchronizations are usually used where rules have changed and need to be applied to existing data, although sometimes people implement unusual configurations that require all objects associated with an MA to be processed. Obviously, this can only be done if the associated performance hit has been found to be acceptable.

Weekly Runs

Many implementations perform a regular, but less frequent, run cycle to perform certain housekeeping tasks. This could be a weekly run, or it could be overnight, for example. Typically, such a run would include clearing the run history up to the last X days, clearing any log files, and performing full imports for those MAs that normally make use of delta imports (to be sure that nothing was missed). This is also a good time to do a backup.

Initial Load and Disaster Recovery Runs

Initial load requirements and disaster recovery requirements are likely to be different, but they do have quite a lot in common. First, they are likely to be performed manually so that the process can be

carefully monitored and any exceptions dealt with as they arise. Secondly, on disaster recovery, even though you may be able to restore a backup database, you will still perform a run cycle very much as though doing a complete reload. The only difference between this and the true initial load is that the authority for some attributes, and even objects, may have changed.

See the topic on disaster recovery for further details on that subject.

Simple Example Script

Here is an example of a very simple batch file that calls a run cycle script every 60 seconds:

```
@ECHO off
SET logfile="C:\SyncScripts\Sync.log"
del /q %logfile% 2> NUL

: Start
ECHO Running Sync Script at %Time% >> %logfile%
ECHO. >> %logfile%

cscript //nologo C:\SyncScripts\SyncFIM.vbs
ECHO.
ECHO Time %Time%
ECHO Sleeping for one minute
ECHO.
timeout /t 60

goto Start
```

Here is the VB script (SyncFIM.vbs) that deletes the run history that is more than two days old and then executes run cycles for MAs:

```
Set Service = GetObject("winmgmts:{authenticationLevel=PktPrivacy}!

root/MicrosoftIdentityIntegrationServer")
Set Server = Service.Get("MIIS_Server.Name='MIIS_Server'")

Wscript.Echo "Clearing run history..."
DeleteDate = DateAdd("d",-2,Now())
DeleteDate = Right("0000" & Year(DeleteDate),4) & "-" & Right("00" &
                          Month(DeleteDate),2) & "-" & Right("00" &
Day(DeleteDate),2)
Server.ClearRuns(DeleteDate)

Set MASet = Service.ExecQuery("select * from MIIS_ManagementAgent where Name = 'HR'")
for each MA in MASet
    Wscript.Echo "Syncing HR Data ..."
    MA.Execute("Export")
    MA.Execute("Full Import")
    MA.Execute("Delta Sync")
next

Set MASet = Service.ExecQuery("select * from MIIS_ManagementAgent where Name = 'FIM'")
for each MA in MASet
    Wscript.Echo "Syncing FIM..."
    MA.Execute("Export")
    MA.Execute("Full Import")
    MA.Execute("Full Sync")
Next
```

```
Set MASet   = Service.ExecQuery("select * from MIIS_ManagementAgent where Name =
'AD'")

for each MA in MASet
    Wscript.Echo "Syncing AD..."
    MA.Execute("Export")
    MA.Execute("Delta Import")
    MA.Execute("Delta Sync")
next
```

Other Run Cycle Considerations

The above example is useful in the classroom, but is not really robust (and 60 seconds is a very short delay).

Stop on Error

It does not stop if there is an error, and if errors occur on import (for example), it might be a bad idea to continue synchronizing and exporting potentially erroneous data to other systems. At the very least, things that should be happening are not happening, but possibly bad things are happening. We should probably stop the run cycle and take a look at the problem.

Checking for Excessive Deletions

After import from a source, if there is any chance that the source could supply erroneous data, you should consider checking the connector space for an unreasonable number of imported deletions because these are the most obvious changes that are likely to be destructive. This is most likely to occur when a data source drops a file for FIM to import, and could possibly drop an incomplete file. If you are performing a Full Import using that file, anything *missing* will be deleted (maybe you have a delta import file, but you think it is a Full Import file). If you find an unlikely number of deletes, you could stop processing any further and take a look (you will need a way of alerting an administrator too).

Here is another command file and associated VB script. Between them, they demonstrate some more principles:

```
@echo off
REM Perform a full import
cscript FullImport.vbs
if {%errorlevel%} NEQ {0} (echo Error[%errorlevel%]: command file failed) & (goto
exit_script)

cscript checkImportDeletions.vbs
if {%errorlevel%} EQU {2} (echo Error[%errorlevel%]: Deletion Ratio Hit - aborting
batch) & (goto exit_script)
if {%errorlevel%} NEQ {0} (echo Error[%errorlevel%]: command file failed) & (goto
exit_script)

cscript DeltaSynch.vbs
if {%errorlevel%} NEQ {0} (echo Error[%errorlevel%]: command file failed) & (goto
exit_script)

REM More commands to perform exports on other systems

:exit_script
```

This runs a full import (the VB script it calls isn't shown here). Then it calls the script below, which checks whether the number of imported deletions is more than 5% of the total number of connector space (CS) objects for the MA in question (an HR MA) . If that figure is exceeded, it does not continue.

```
dim managementagent
dim ratio
dim result

ratio = 5
Set wmiService = GetObject("winmgmts:{authenticationLevel=PktPrivacy}!
                                        root/MicrosoftIdentityIntegrationServer")
Set managementagent = wmiService.Get( "MIIS_ManagementAgent.Name='HR'")

result = managementagent.NumImportDelete / managementagent.NumCSObjects * 100
if result >= ratio then
        WScript.Echo "%ATTENTION: Ratio is hit. Setting errorlevel to 2"
        Wscript.quit(2)
end if

WScript.Echo "%Success: Ratio is not hit"
Wscript.quit(0)
```

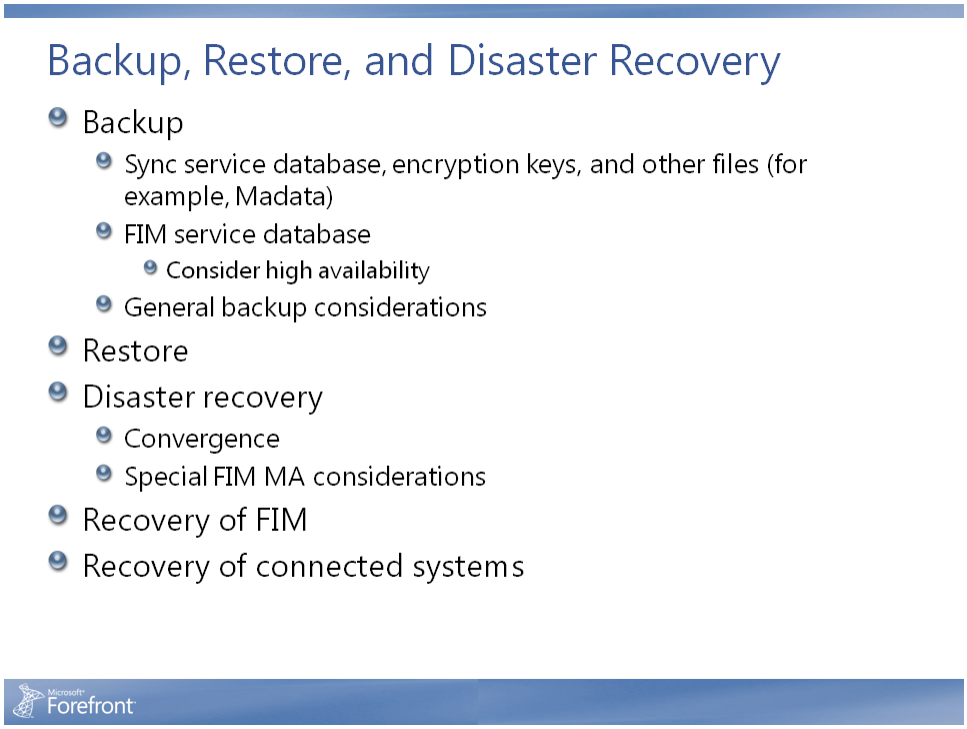
Avoiding Unnecessary Processing

Complex systems may have many MAs connected to data sources, and additional MAs for handling requirements, such as group or role management and data transformations. If no changes come in during an import, it might be time-wasting to go through the rest of activity, and you could choose to stop processing until the next run cycle. The above script could be adapted to detect no import changes by making use of NumImportAdd, NumImportDelete, and NumImportUpdate. A similar script might be used for export, using NumExportAdd, NumExportDelete, and NumExportUpdate.

Alerting an Administrator

If something goes wrong, the usual method for alerting an administrator (operator) is to use Microsoft Operations Manager (MOM) or System Center Operations Manager (SCOM) to pick up a logged event (either logged anyway because of the error, or logged by your script).

Backup, Restore, and Disaster Recovery



Backup

Backup is primarily about backing up the two databases, but there are a few other bits and pieces.

The FIM Synchronization Service

Database

When you back up the database, you are backing up:

- Identity data
- The synchronization service configuration data
- The contents of the extensions folder

Additionally, you should back up:

- Encryption keys (only if they change)
- Any important files in the Madata folder and sub-folders (these would normally be import and export files for file-based MAs, but might include templates)
- Any source code for extensions

Synchronization Service Configuration

When you back up the FIM synchronization service database, you include data and configuration and even the contents of the extensions folder. However, it can be useful to save (and load) just the configuration, which can be done using the **Export Server Configuration…** and **Import Server**

Configuration … options from the File menu in the synchronization service manager. Note that the server export includes an XML for each MA (the same as if they had each been exported) and another for the metaverse.

Although initial migration from test to production might be performed using a Microsoft® SQL Server® backup and restore, subsequent changes can be migrated using server export because it preserves data in the target (production) system. This set of XML files also represents an ultimate backup—in a convergent system you can rebuild the metadirectory from just this configuration (by importing and synchronizing all the data again).

Export/Import Update Management Agents

An MA configuration can be exported, creating an XML file. That file can then be imported into a different system to create a new MA, or it can be used to update an existing MA's configuration. Uses of this include:

- Migrating MAs from test to production (export/import)
- Migrating updates from test to production (export/update)
- Re-use of MA configuration (where one connected source is very similar to another)
- Sending configuration to someone to help debug problems

Export/Import Schema

You can export/import the metaverse schema to/from an XML file using the appropriate options on the actions file menu of the metaverse designer in synchronization service manager. An obvious use of this is to migrate a schema change from test to production.

> **Note:** The schema plus all the MAs almost add up to the entire configuration, but not quite. What is missing is the precedence relationships between the MAs.

Encryption Keys

The encryption keys are normally backed up during installation, but you can use the key management utility (miiskmu.exe) to back them up again (and generate new keys). Note that only passwords are encrypted (not general identity attributes): MA passwords, passwords on new accounts provisioned but not yet exported, PCNS passwords queued up waiting for a connection to a server, and extensible MA parameters that are configured as encrypted.

The backed up keys are needed if you want to use the data with a fresh installation (for example, after disaster recovery), but the consequences of losing the keys are only the re-entry of MA passwords, some full syncs (to re-provision accounts), and for some users to change passwords again to get them back in sync.

The FIM Service

As with the synchronization service database, the configuration is held along with the data, but you should be careful to think through what other files might be needed, for example, workflow source and object files. You will also have a range of Windows PowerShell and VBScript scripts for the various

management activities that are required. It is impossible to give an exhaustive checklist of items to back up here, but consideration must be given to all custom components.

High Availability

Loss of the FIM service database will generally be a more serious matter than loss of the synchronization service database. In a well-formed, convergent implementation, the metadirectory database can be rebuilt as long as you have the configuration. But the FIM service database contains a lot of proprietary, authoritative data, and should be treated more like a source (like HR or AD). In fact you might consider the loss of this database such a serious matter, and its 100% recovery so difficult to achieve, that it is worth investing heavily in a high availability solution, or at least implementing a full recovery SQL Server model rather than a simple backup.

General Backup Considerations

Remember that backup truncates your log file. If you don't back up often, for example in a development environment, you can struggle with disk space on the log files partition.

A complex FIM implementation may also use other databases to support it (for example, for certain types of group or role management, or supporting delta import from non-delta sources). These should be backed up too.

FIM Service Configuration

While the synchronization service manager has options for exporting and importing configuration, it is done rather differently in the FIM service.

FIM ships with some Windows PowerShell cmdlets primarily intended for exporting configuration data, comparing it, and importing it. So, for example, you can migrate configuration changes from a test or pilot system to a production system by exporting both configurations, compare them to identify the differences, and then import the differences to the production system. The cmdlets are:

Command	Purpose
Export-FIMConfig	To export objects
ConvertFrom-FIMResource	To write the exported data to an XML file
ConvertTo-FIMResource	To load exported data from an XML file
Join-FIMConfig	To match up objects between two exports (to find out what is common between them)
Compare-FIMConfig	To identify the differences
Import-FIMConfig	To import the differences

To use these cmdlets, you must run Windows PowerShell and use the Add-PSSnapin command to load them using:

```
Add-PSSnapin FIMAutomation
```

When you use the cmdlets, they call methods in the ResourceManagementService DLL (you can call these directly if you like).

For help on the cmdlets, type:

```
get-help <cmdlet name>
```

This is the province of a more advanced course, but you can find information about the cmdlets on TechNet: http://technet.microsoft.com/en-us/library/ee534906(WS.10).aspx.

Restore

On the face of it, you are restoring two databases (plus any other supporting databases), and a few folders. If you are merely migrating an empty system from test to production, that is fine, but if this is a disaster recovery situation, there is rather more to it. See the next section.

In any case, in order to restore the databases, the relevant FIM services will have to be stopped. When restarting the services, in the case of the FIMSynchronizationService, you should use the miisactivate.exe command (usually found in C:\Program Files\Forefront Identity Manager\2010\Synchronization Service\Bin\) to restart the service, in order for the extension files to be treated correctly. Something like:

```
miisactivate "[the path and name of keys file]"  Litware\FIMSync Pa$$w0rd
```

Disaster Recovery

Usually if you are using a restore, you are recovering from a disaster, however minor (an obvious exception is when migrating an entire configuration). The chances are that your backup is at least a few hours old, and maybe much more, but the connected systems will have changed since the backup was taken. It is important, therefore, that the latest data is imported and synchronized before normal service resumes.

In fact, the process you go through is not a lot different whether you have restored a backup database, or are doing a complete rebuild (where you have the configuration, but no data). The key issue is that existing data must be properly connected and synchronized before the system starts provisioning again.

Convergence

The issues involved in non-convergence are the province of a more advanced course. However, it is important to mention in brief that a convergent system is one in which, given the same inputs, the data in the system arrives at the same state, regardless of the order in which the processes in the system are executed. In practice, this means that during a recovery, if your system is convergent, you can import your data and run your synchronizations in any order you like, and your system will end up in the desired state.

To understand this well, we have to go into a little history. Installations of ILM (the predecessor of FIM) are often convergent—the major exceptions to this are caused by any manual actions, manual

precedence rules, explicit connector and disconnectors, and “do not recall attributes”. The synchronization service is more or less the same as ILM, but FIM introduces a few more non-convergent features:

- Equal Precedence can cause a system to be non-convergent. Because attributes set to equal precedence are last-sync-wins attributes, you must be careful to synchronize the contributing sources in an appropriate order. Of course, during a recovery, it will probably be impossible to recreate the precise order in which the individual attribute values were modified in the external systems (that is, a true Last-Write-Wins), and so the recovered system will be a best effort solution (and this illustrates the specific issue which many FIM and ILM designers have with systems involving Last-Write-Wins scenarios).
- The special nature of the FIM MA also throws up some challenges. Let us assume that you have a system that is broadly convergent, or that you have at least accepted the limitations of your level of convergence. There is one major exception to consider—the provisioning into the FIM MA cannot be switched off.

Consider the case in which the synchronization service has failed, corrupting its database. A backup is available, but since the backup was taken, changes have been made in a variety of connected systems, including the FIM portal. If our first import is from an authoritative data source other than the FIM portal, because FIM portal provisioning cannot be turned off, provisioning will start to take place into the portal even though corresponding objects already exist, potentially creating duplicates or other data conflicts. It is important that an import is executed on the FIM service MA first, so that any provisioning into the portal hits the existing objects (where they exist), and fails.

But perhaps we are missing an even more obvious reason for importing from the FIM portal first—we need any declarative synchronization rules to be present!

> **Note:** Even if your system is broadly convergent, it is important that you synchronize the FIM service MA first.

Exporting to FIM

As just mentioned, we may need to provision into FIM during disaster recovery, but perhaps this would not normally happen because the FIM portal is authoritative for object creation, but flows and precedence rules may be such that export is not facilitated.

For example, suppose that during normal operation, user objects are created in the FIM portal and exported to (provisioned into) AD. Flows and precedence are all in that direction. On disaster recovery, we could find ourselves with users in AD whose counterparts in the FIM portal have been lost. We must now either reconfigure flow and precedence, project the AD users into the metaverse, and provision them into the FIM portal; or we can manually add them in the portal, turn off provisioning, project them into the metaverse, and use a join rule in the AD MA to connect them up.

Actually, the point should be made that what we are talking about applies equally to any authoritative source. If, for example, the HR system is damaged, and a backup has to be restored, missing objects would have to be treated in much the same way—manually added and then joined up. The only difference is that the FIM MA join rules are built-in (and based on MVGUID), whereas you can configure whatever join rules you need in an HR MA, giving you more flexibility.

Recovery of FIM

In summary, taking all of the above into account:

1. Equal precedence may have introduced non-convergence (or further non-convergence). There is no easy answer to this.
2. We must consider the FIM MA as a special case. It should be synchronized first.
3. It could be that we need to make some configuration changes, or manual entries, in order to accommodate missing objects.

A typical approach might be:

1. If necessary, make adjustments to configuration.
2. Switch off both kinds of provisioning.
3. Run a full import run profile from all sources.
4. Perform a synchronization for the FIM MA (you could perform a delta synchronization, but a full synchronization is more appropriate in this case). This will probably project a lot of objects, but not provision any.
5. Perform synchronizations for all other MAs that are authoritative for object creation. These may find some joins and may project more objects, but not provision any.
6. Perform a synchronization for all other MAs, so that connectors can join to the objects that have been projected.
7. Export any changes, import to confirm the changes, and synchronize. Repeat until things stop changing.
8. If necessary, examine the CS for disconnectors that may need manual entry in authoritative systems, add them, and repeat the above steps.
9. Switch on provisioning and perform full synchronizations to fill in the gaps where a CS object should exist, but doesn't.
10. Export any changes, import to confirm the changes, and synchronize. Repeat until things stop changing.
11. If necessary, reconfigure back to the normal way of doing things and go through an entire manual cycle before engaging your regular script.

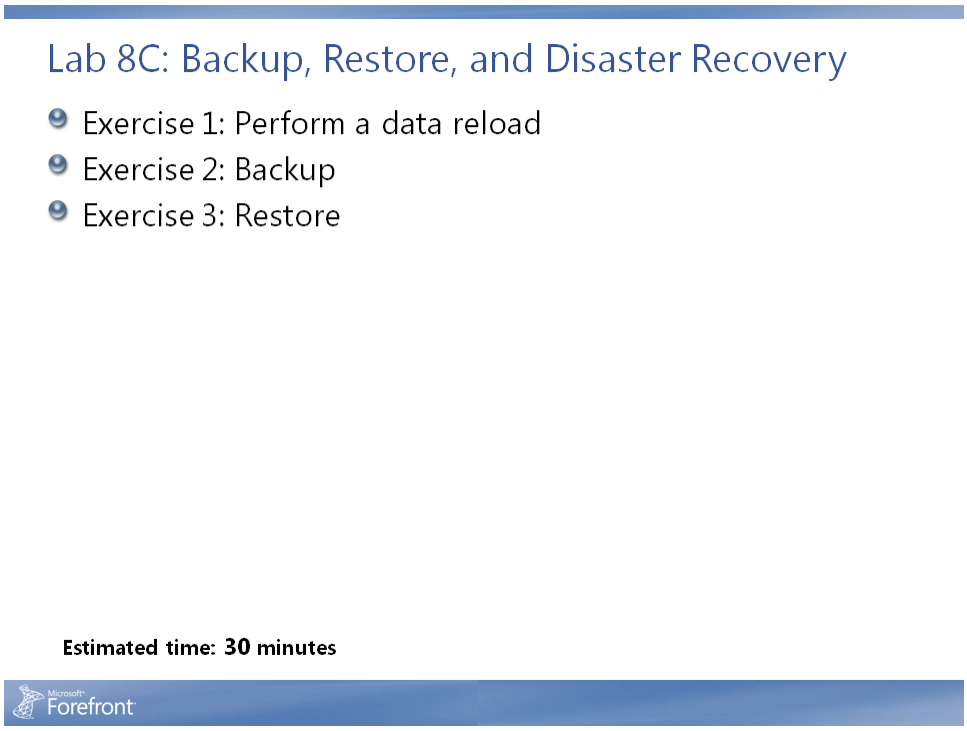
Recovery of Connected Systems

It is worth mentioning that if a connected system fails (only), and is restored from a backup that is not absolutely current, all sorts of changes may find their way back to FIM, including the reappearance of accounts, and unexpected transitions could occur, firing MPRs. So while we might not normally plan for the reappearance of accounts for someone who has left the company (and, therefore, have MPRs to deal with this case), if a system is restored to a former state, this might be exactly the case we have to deal with! It is unlikely that you will make technical plans for every one of these possible cases, but you

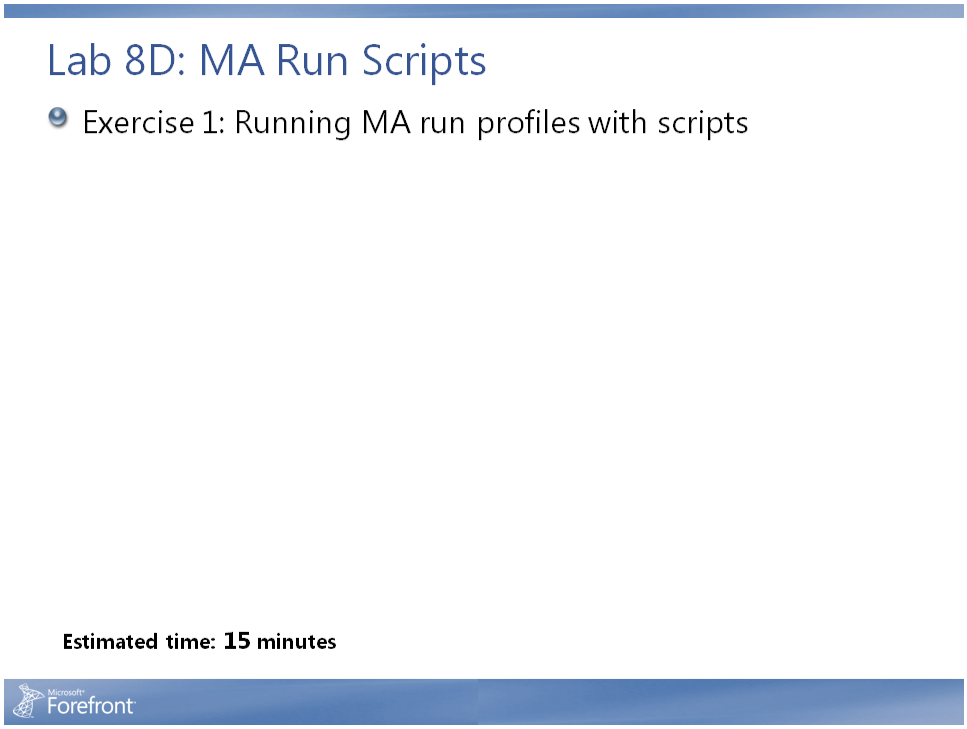
should at least bear them in mind when making organizational arrangements. The FIM team should at least be informed if one of their connected systems is being restored.

> **Note:** You should have a disaster recovery plan and you should test it (within reason). Even if you have a tested plan, every disaster recovery is different, and you should be perform it slowly and after much consideration of the right approach and possible consequences—with a special thought for any points of no return.

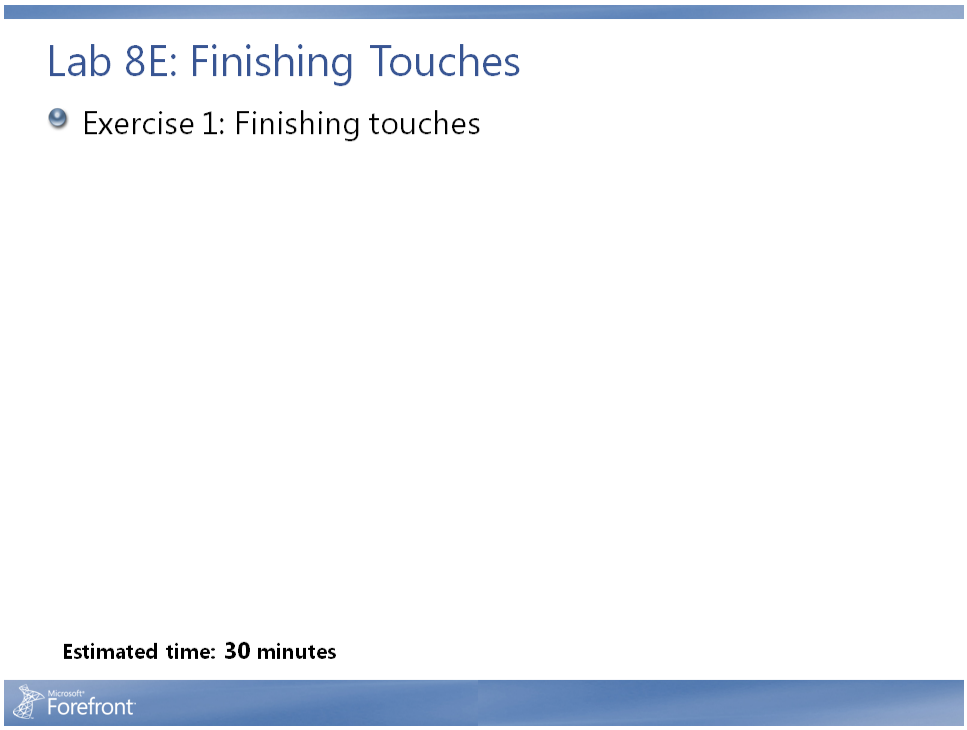
Lab 8C: Backup, Restore, and Disaster Recovery



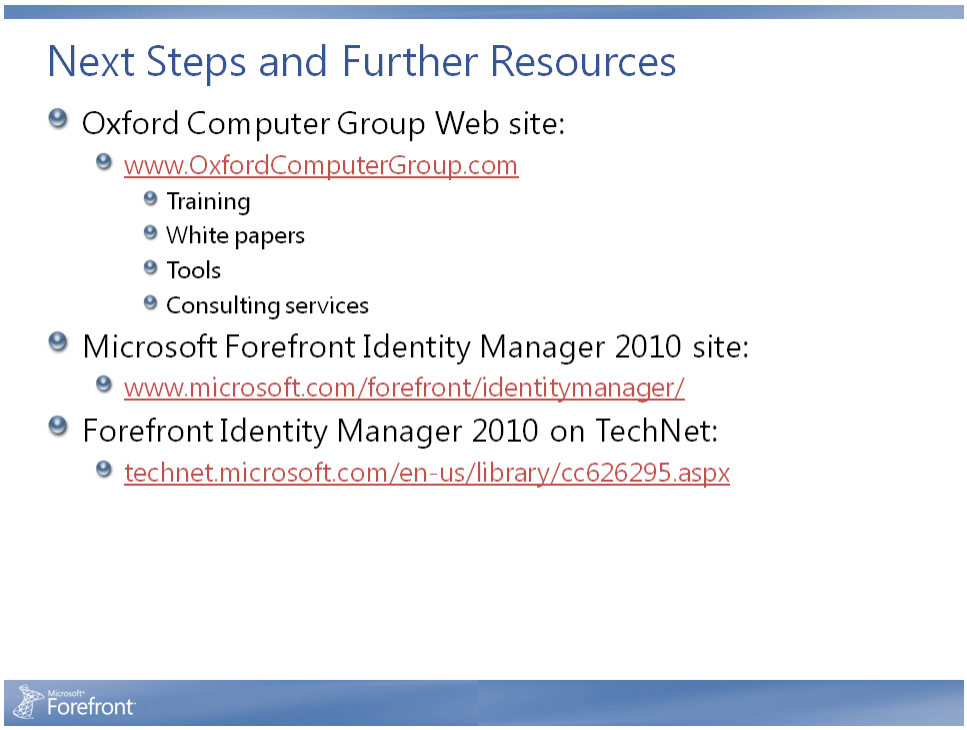
Lab 8D: MA Run Scripts



Lab 8E: Finishing Touches



Next Steps and Further Resources



To excel with FIM, you must make time to continue learning:

- Visit www.OxfordComputerGroup.com where you can access classroom and online training, white papers, custom tools, and consulting services.
- Watch the Courseware Download Center Web site and TechNet for additional courses, Microsoft Press books, white papers, and technical information.
- Monitor the Microsoft Forefront Identity Manager 2010 site for updates on FIM.