

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ

Федеральное государственное бюджетное образовательное учреждение
высшего образования

«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Политехнический институт
Факультет приборостроения,
информационных технологий и
электроники

Кафедра
«Информационная безопасность
систем и технологий»

Специальность 100503 –

Специализация -

Информационная безопасность
автоматизированных систем
Защищенные автоматизированные
системы управления

ДИПЛОМНАЯ РАБОТА

на тему

ПРАКТИКУМ ПО ПРИМЕНЕНИЮ МЕТОДОВ И СПОСОБОВ ЗАЩИТЫ
КОМПОНЕНТОВ ВИРТУАЛЬНОЙ ИНФРАСТРУКТУРЫ

Студент	_____	<u>Шилов Богдан Сергеевич</u>
Руководитель	_____	<u>Фатеев А.Г.</u>
Нормоконтролёр	_____	<u>Фатеев А.Г.</u>
Рецензент _____	_____	<u>Зорин А.Н.</u>

Работа допущена к защите (протокол заседания кафедры от _____ № _____)

Заведующий кафедрой _____ С.Л. Зефилов

Работа защищена с отметкой _____ (протокол заседания ГЭК
от _____ № _____)

Секретарь ГЭК _____ А.П. Иванов

Пенза 2017

СОДЕРЖАНИЕ

Обозначения и сокращения.....	6
Введение.....	7
1 Применение оболочки Windows PowerShell для управления серверами виртуализации.....	9
1.1 Развитие оболочки Windows PowerShell и сравнение с другими оболочками.....	9
1.2 Оболочка командной строки cmd.exe.....	12
1.3 Сервер сценариев Windows Script Host (WSH).....	14
1.4 Оболочка WMI Command-line (WMIC).....	16
1.5 Причины и цели создания оболочки PowerShell.....	17
1.6 Отличие PowerShell от других оболочек — ориентация на объекты.....	18
2. Описание существующих платформ и технологий виртуализации.....	19
2.1 Определение виртуализации.....	19
2.2 Виды виртуализации.....	19
2.2.1 Аппаратная виртуализация.....	19
2.2.2 Программная виртуализация.....	20
2.2.3 Прочие виды виртуализации.....	20
2.3 Платформы виртуализации.....	21
2.3.1 Microsoft Hyper-V	21
2.3.2 VMware.....	22
2.3.3 Citrix Xen.....	23
2.3.4 KVM.....	24
2.3.5 Платформа, используемая при разработке практикума.....	25
3 Описание подготовки виртуальных стендов, используемых при выполнении заданий практикума.....	26

3.1 Установка ПО VirtualBox и компонента Extension Pack.....	26
3.2 Создание виртуальных машин.....	29
3.3 Установка гостевых ОС.....	32
3.4 Настройка виртуальной локальной сети.....	33
3.5 Настройка сетевых подключений гостевых ОС.....	35
3.6 Проверка работы оболочки PowerShell.....	37
3.7 Результат создания виртуального стенда.....	37
4 Порядок выполнения задания по изучению применения политик выполнения PowerShell для управления запуском сценариев.....	39
4.1 Цель задания.....	39
4.2 Создание сценариев с использованием PowerShell.....	39
4.3 Настройка политик на локальном сервере	41
4.4 Настройка политик на удаленном сервере.....	44
5 Порядок выполнения задания по изучению использования сертификатов для управления запуском сценариев PowerShell.....	47
5.1 Цель задания.....	47
5.2 Подписывание сценариев с помощью утилиты makecert.....	47
5.3 Создание сертификатов с помощью встроенных командлетов PowerShell.....	54
6 Описание порядка выполнения заданий практикума и результатов выполнения заданий.....	59
6.1 Порядок выполнения заданий практикума.....	59
6.2 Порядок выполнения задания по изучению применения политик выполнения PowerShell для управления запуском сценариев.....	59
6.2.1 Начало выполнения задания практикума, подготовка сценария.....	59
6.2.2 Проверка настроек политики выполнения сценариев на локальном сервере	60

6.2.3 Проверка настроек политики на удаленном сервере.....	60
6.2.4 Описание порядка выполнения задания в виде UML-диаграммы	61
6.3 Порядок выполнения задания по изучению использования сертификатов для управления запуском сценариев PowerShell.....	61
6.3.1 Начало выполнения задания практикума, подготовка сценария.....	61
6.3.2 Подписывание сценариев с помощью утилиты makecert.....	61
6.3.3 Проверка настроек политики подписанного сценария на локальном сервере	62
6.3.4 Проверка настроек политики на удаленном сервере....	62
6.3.5 Подписывание сценариев с помощью штатных командлетов оболочки Windows PowerShell.....	62
6.3.6 Описание порядка выполнения задания в виде UML-диаграммы	63
7 Описание структуры и содержания отчета о выполнении практических заданий.....	64
7.1 Общий перечень данных, содержащихся в отчете.....	64
7.2 Структура отчета по выполнению задания по изучению применения политик выполнения PowerShell для управления запуском сценариев.....	64
7.3 Структура отчета по выполнению задания по изучению использования сертификатов для управления запуском сценариев PowerShell.....	65
8 Определение порядка контроля выполнения практического задания.....	66
8.1 Общий порядок контроля.....	66

8.2 Порядок контроля задания по изучению применения политик выполнения PowerShell для управления запуском сценариев.....	66
8.3 Порядок контроля задания по изучению использования сертификатов для управления запуском сценариев PowerShell.....	68
9 Подготовка сценария для автоматизированного развертывания виртуальных машин.....	72
Заключение.....	73
Список использованных источников.....	75
Приложение А Состав виртуального стенда.....	75
Приложение Б UML–диаграмма последовательности действий выполнения практического задания по применению политик выполнения PowerShell для управления запуском сценариев.....	77
Приложение В UML–диаграмма последовательности действий выполнения практического задания по использованию сертификатов для управления запуском сценариев PowerShell	79
Приложение Г Структура отчета о выполнении заданий практикума...	81
Приложение Д Контрольный экземпляр отчета по выполнению задания по изучению применения политик выполнения PowerShell для управления запуском сценариев	83
Приложение Е Контрольный экземпляр отчета по выполнению задания по изучению применения политик выполнения PowerShell для управления запуском сценариев.....	86
Приложение Ж Порядок проверки выполнения заданий практикума....	90
Приложение И Сценарии для автоматизированного развертывания ВМ..	92

ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ

- ARM - Advanced RISC Machine (усовершенствованная RISC-машина);
- CIM - Common Information Model (общая информационная модель);
- DOS - Disk Operating System (дисксовая операционная система);
- HTML - Hyper Text Markup Language (язык гипертекстовой разметки);
- MS-DOS - Microsoft Disk Operating System (дисксовая операционная система от Microsoft);
- USB - Universal Serial Bus (универсальная последовательная шина);
- VHD – Virtual Hard Disk (виртуальный жесткий диск);
- WMI - Windows Management Instrumentation (инструментарий управления Windows);
- WMIC – Windows Management Instrumentation Command-line (инструментарий управления Windows в командной строке);
- WSH –Windows Script Host (сервер сценариев Windows);
- WQL - WMI Query Language (язык запросов WMI);
- XML - eXtensible Markup Language (расширяемый язык разметки);
- XSL - eXtensible Stylesheet Language (расширяемый язык стилей);
- ВКР – выпускная квалификационная работа;
- ВМ – виртуальная машина;
- ОЗУ – оперативное запоминающее устройство;
- ОС – операционная система;
- ПК – персональный компьютер;
- ПО – программное обеспечение.

ВВЕДЕНИЕ

В последнее время технология виртуализации применяется для решения все большего количества различных задач. Это объясняется тем, что технология виртуализации имеет преимущества, а именно позволяет экономить физические и аппаратные ресурсы, так как вместо физических (аппаратных) серверов используются виртуальные машины. Однако применение виртуализации порождает и ряд проблем, в том числе проблемы безопасности виртуальных инфраструктур.

Для управления виртуальными инфраструктурами могут применяться различные средства. Как правило, это графические консоли, входящие в состав ОС, например Windows Server 2012 R2, либо средств управления платформ виртуализации, например VMware vSphere. Однако может возникать ситуация, когда эти консоли не установлены, например, когда на сервере виртуализации установлен только гипервизор. В таком случае для управления применяются средства, реализующие интерфейс командной строки. Одним из таких средств является оболочка Windows PowerShell, которая применяется во многих платформах виртуализации.

Оболочка PowerShell является удобным средством управления, реализующим большой набор функциональных возможностей по управлению ВИ и отдельными ее компонентами. В данной оболочке также присутствует возможность автоматизации действий администратора с помощью сценариев PowerShell и при использовании планировщика задач. При использовании оболочки PowerShell необходимо учитывать проблемы обеспечения безопасности ВИ. Например, несанкционированный запуск сценариев на серверах виртуализации. Для противодействия таким угрозам могут применяться встроенные механизмы обеспечения безопасности, которые рассмотрены в рамках ВКР.

Актуальность ВКР обусловлена тем, что администраторы ВИ должны уметь не только применять оболочку PowerShell для управления, но и уметь

применять механизмы обеспечения безопасности. Такая ситуация может возникнуть при отсутствии специальных средств защиты информации или при невозможности их использования для обеспечения безопасности ВИ.

В ВКР был разработан практикум, который включает два задания, посвященных изучению механизмов контроля выполнения сценариев, а также использованию сертификатов для подписания сценариев и контроля выполнения подписанных сценариев.

Для изучения этих механизмов безопасности используется лабораторный стенд, включающий несколько ВМ, объединенных в локальную сеть, которая имитирует ВИ.

Для выполнения задания ВКР были подготовлены ВМ, входящие в состав лабораторного стенда, и составлены описания порядка выполнения двух заданий, входящих в состав практикума.

1 Применение оболочки Windows PowerShell для управления серверами виртуализации

1.1 Развитие оболочки Windows PowerShell и сравнение с другими оболочками

В настоящее время многие пользователи и начинающие администраторы настолько привыкли к графическому интерфейсу Windows, что даже перестали задумываться о том, что управление может осуществляться не только нажатиями кнопками мыши по экрану, но и с помощью командной строки и различных сценариев. Чаще всего пользователи просто не догадываются, в чем заключаются преимущества, присущие этим инструментам для автоматизации работы.

Данное положение объясняется тем, что исторически командная строка в операционной системе всегда была уязвимым местом. Основной причиной этого является, прежде всего, то, что изначально компания Microsoft была ориентирована на широкий круг пользователей, которые не имеют особых навыков в администрировании операционных систем, а значит не желающих углубляться в технические детали выполнения тех или иных задач в системе. Все это и привело к тому, что разработчики операционной системы делали акцент на модернизации и оптимизации графической части системы, что позволяло повысить комфорт пребывания пользователей за компьютером, вместо того, чтобы делать шаги в сторону создания рабочего окружения для опытных пользователей и специалистов.

С течением времени стало понятно, что стратегия акцентирования внимания на графической части принесла успех компании с коммерческой точки зрения на рынке персональных, домашних, а также офисных компьютеров, ведь миллионы людей используют графический интерфейс для запуска нужных им программ, работы в офисных пакетах, просмотра фильмов, а также прочих взаимодействий с операционной системой. Да и на данный момент управлять одним сервером на операционной системе Windows не

представляет никакого труда, а все из-за того, что операционная система имеет множество удобных графических решений, позволяющих без затруднений настроить различные параметры, а также выполнить ежедневные задачи. Более того, служба терминалов позволяет обеспечить удаленный доступ к другому компьютеру, а значит, работа на удаленном сервере не является затруднением, независимо от его расположения.

Удобство управления одним сервером не означает, что несколькими серверами будет управлять так же удобно, т.к. подобная модель управления не является масштабируемой. Это значит, что для изменения настроек на десяти серверах придется затратить в десять раз больше времени, потому что последовательность необходимых действий в диалоговых окнах придется повторить десять раз. Всё это означает, что вопрос автоматизации рутинных операций, таких как мониторинг работы служб, анализ журналов событий, инвентаризация программного обеспечения и оборудования организации, является наиболее актуальным в процессе администрирования множества компьютеров. Решить данную проблему можно либо с помощью специальных программ, которые, как правило, занимают много места на жестком диске, а также стоят немалых денег, либо сценарии, которые пишутся администраторами самостоятельно либо на языке оболочки командной строки, либо на языке одной из специальных оболочек, предназначенных для администрирования. Сценарии, в большинстве своем, поддерживаются непосредственно операционной системой, без установки сторонних программных продуктов.

Все это означает, что специалисту, который занимается администрированием информационных систем на базе операционной системы Windows, знание возможностей командной строки, принципов написания сценариев и умелое владение технологиями автоматизации, поддерживаемых операционной системой, просто необходимо.

Учитывая всё вышесказанное можно сделать вывод, что сценарии и командная строка необходимы только администраторам. Но ведь можно сократить до минимума время, затрачиваемое на выполнение многих повседневных задач, таких как копирование или архивирование файлов, подключение или отключение сетевых ресурсов, в выполнении которых обычно помогает графический интерфейс проводника Windows. Полная автоматизация достигается лишь написанием простого файла-сценария, в котором необходимая последовательность действий будет выполняться небольшим набором команд. К сожалению, большинству пользователей это не по силам, ведь даже основные команды и базовые возможности, такие как перенаправление ввода/вывода и конвейеризация команд, им не по силам. Например, формирование файла, в котором будут содержаться имена файлов всех подкаталогов, для обычного пользователя может показаться трудоемкой задачей, которая отнимет много времени. Если знать необходимую команду для командной строки, то данная задача превратится во ввод одной команды `dir` с несколькими ключами и дальнейшим перенаправлением вывода этой команды в текстовый файл.

Вопрос безопасности – наиболее важный вопрос в любом программном обеспечении, и PowerShell в этом случае не исключение. Меры защиты PowerShell не обеспечат абсолютную защиту, но предотвратить некоторые угрозы они все-таки смогут.

Первая мера безопасности – невозможность запуска сценария только по имени, даже находясь в папке, где он находится. Это исключает возможность перехвата команды вредоносными сценариями. Например, для запуска сценария с названием `script1.ps1`, находясь при этом в папке `C:\scripts`, необходимо указать полный путь к сценарию, иначе оболочка выдаст сообщение об ошибке.

Кроме этого существует еще одна мера безопасности – политика выполнения сценариев. Существует множество значений этой политики, но основными являются следующие:

- «Restricted» - конфигурация PowerShell по умолчанию. Эта конфигурация не допускает запуска даже подписанных сценариев. Существует лишь возможность запускать одиночные команды напрямую;

- «AllSigned» – конфигурация, позволяющая выполнять сценарии PowerShell, которые имеют электронную подпись от доверенного источника. Перед выполнением сценария выводится предупреждение;

- «RemoteSigned» – конфигурация, позволяющая выполнять сценарии PowerShell, загруженные из Интернета при условии, что они имеют электронные подписи от доверенного источника. Локальным сценариям в этом случае подпись не требуется. Запуск сценария не сопровождается предупреждениями, однако о сценариях, которые хоть и подписаны, но потенциально опасны, предупреждение выводится;

- «Unrestricted» – конфигурация, позволяющая запуск без предупреждения абсолютно всех сценариев, включая неподписанные, скачанные из Интернета и т.п.

1.2 Оболочка командной строки cmd.exe

Интерактивная оболочка командной строки поддерживается во всех версиях Windows, определенный набор утилит командной строки устанавливается по умолчанию, но состав и количество этих утилит зависит уже от версии ОС. Вообще, любая ОС представляется в виде совокупности ядра системы, которое имеет доступ к аппаратуре и управляет файлами и процессами, и командного интерпретатора (оболочки) с утилитами, обеспечивающие пользователю доступ к функциональности ядра ОС. В разных ОС механизм работы оболочек ничем не отличается: оболочка выводит приглашение на ввод команды пользователю, пользователь вводит какую-либо команду, которая реализуется либо непосредственно самой оболочкой, либо

определенной внешней утилитой, далее оболочка выполняет введенную команду, после чего выводит приглашение на ввод следующей команды. Оболочка поддерживает не только интерактивный режим работы, в котором команды вводятся последовательно пользователем вручную, существует также и пакетный режим, который, как правило, поддерживается многими оболочками. В нем команды выполняются последовательно, но ввод происходит не с клавиатуры, а из файла-сценария, в котором эти команды последовательно записаны.

Привычная оболочка командной строки `cmd.exe` впервые появилась в Windows NT, она представлялась интерпретатором, который расширял возможности оболочки `command.com`, которая была в ОС MS-DOS. Командный интерпретатор `command.com`, в свою очередь, свою функциональность позаимствовал из операционной системы CP/M, где командный интерпретатор представлял собой значительно урезанный и упрощенный вариант оболочки UNIX-систем.

Таким образом, оболочка командной строки `command.com` в MS-DOS всегда уступала оболочкам UNIX систем, как по развитости языка сценариев, так и по удобству работы. В командной оболочке `cmd.exe` в Windows NT также не удалось улучшить эргономику оболочки, несмотря на все сделанные улучшения: в режиме интерактивной работы все также отсутствовала поддержка псевдонимов для длинных названий команд, автоматическое завершение команд было всё также не реализовано, синтаксис и возможности языка командных файлов тоже отставали.

Более того, еще одним недостатком оболочек от Microsoft было то, что функциональные возможности стандартных, не требующих установки дополнительных компонентов, утилит командной строки было очень ограничено. В то время как у оболочек UNIX-систем таких возможностей было намного больше. Та же ситуация прослеживается и со встроенной справочной

системой оболочек от Microsoft, по качеству и полноте справочных систем UNIX-оболочки все также превосходят оболочку cmd.exe.

Таким образом, оболочка командной строки cmd.exe от Microsoft и ее командные файлы являются наиболее универсальными и простыми для изучения средствами автоматизации работы в ОС Windows, которые доступны во всех версиях ОС. Однако эта оболочка и ее командные файлы существенно проигрывают по функционалу и эргономике аналогичным оболочкам в UNIX-системах и не обеспечивают доступ к объектным моделям, поддерживаемым операционной системой (COM, WMI, .NET).

1.3 Сервер сценариев Windows Script Host

Следующим этапом развития средств и технологии автоматизации в ОС Windows стало появление сервера сценариев под названием Windows Script Host (WSH). Этот инструмент был разработан для всех версий ОС Windows, позволяющий выполнять сценарии уже на полноценных языках сценариев, таких как VBScript и Jscript, непосредственно в ОС. До появления инструмента эти языки были доступны только внутри HTML-страниц и работали в контексте безопасности веб-браузеров, из-за этого эти сценарии не имели доступа к файловой системе локального компьютера

Инструмент WSH имеет ряд преимуществ относительно командных файлов оболочки cmd.exe.

Первое преимущество заключается в том, что VBScript и JScript являются полноценными алгоритмическими языками, а значит они имеют встроенные функции и методы обработки символьных строк, возможность выполнения математических операций, возможность обработки исключительных ситуаций и т.п. Более того, в WSH имеется возможность использовать не только встроенные языки сценариев, но и любой другой, например Perl, достаточно установить соответствующий модуль поддержки.

Вторым преимуществом является то, что WSH имеет поддержку нескольких собственных объектов, свойств и методов, которые позволяют

решать множество рутинных задач администраторов ОС, например работа с системным реестром, сетевыми ресурсами, переменными средами, ярлыками и специальными папками Windows, а также запуск и управление работой сторонних приложений.

Третье преимущество заключается в том, что к службам любых приложений-серверов автоматизации, которые регистрируют в операционной системе свои объекты, можно обратиться напрямую из сценариев WSH

Еще одним преимуществом является то, что сценарии WSH имеют возможность работы с объектами информационной модели Windows Management Instrumentation, которая обеспечивает программный интерфейс управления всеми компонентами операционной модели, а также с объектами службы каталогов Active Directory Service Interface.

Кроме преимуществ, как и прочее ПО, WSH имеет ряд ограничений. Прежде всего, WSH является только средой выполнения сценариев, а не полноценной оболочкой. Это значит, что WSH не интегрирован с командной строкой, то есть режим, в котором можно было бы вводить команды с клавиатуры и сразу видеть результат и выполнения, отсутствует.

Еще одним существенным ограничением является то, что по умолчанию в ОС нет полноценной справочной информации о WSH. Это говорит о том, что всю документацию необходимо искать предварительно в Интернете на официальном сайте Microsoft. Все это значит, что при написании сценария необходимо точно знать синтаксис команды, либо точное название объекта WSH, с которым необходимо произвести манипуляции. Иначе не имея распечатанной документации, либо не имея устройства с выходом в Интернет для получения справки, написать корректный сценарий не удастся.

В итоге, можно сказать, что сценарии WSH является универсальным инструментом, который позволяет решать задачи автоматизации практически любой сложности в любой версии ОС, однако для этого требуются соответствующие навыки в работе с данным инструментом, а также

необходимо знание самих языков сценариев и ряда смежных технологий управления ОС.

1.4 Оболочка WMI Command-line (WMIC)

Как сказано выше, в ОС Windows есть поддержка информационной модели Windows Management Instrumentation (WMI), занимающей весомое место среди инструментов, технологий и средств автоматизации. Основой для данной технологии является схема Common Information Model (CIM), представляющая собой логическую и физическую структуры компьютерных систем в образе единой расширяемой объектно-ориентированной информационной модели и определяющая единые интерфейсы для получения какой-либо информации о каждом компоненте этой модели.

Сначала работа с WMI была возможна двумя способами: при помощи составления сценариев высокой сложности, либо при помощи специальных графических утилит. Утилита WMI Command-line, которая позволяет совершать обращение к подсистеме WMI прямо из командной строки, была включена в состав ОС Windows Server 2003 и ОС Windows XP. С помощью оболочки WMIC можно совершать навигацию по информационной схеме WMI как удаленно, так и локально, позволяя при этом выполнение WQL-запросов к объектам и классам WMI. В это же время оболочка поддерживает псевдонимы, которые можно использовать вместо сложных названий классов WMI. Также имеется возможность создания собственных псевдонимов, а значит информационная схема WMIC является расширяемой.

Изначально оболочка WMIC имеет поддержку около 80 псевдонимов, которые позволяют выполнять более ста методов, а также получить значения множества свойств. Еще одно преимущество оболочки WMIC заключается в том, что вывод команд возможен разными способами: в обычный текстовый файл, в HTML/XML-документ, в .mof файл, на экран монитора, в текстовый файл с разделителями, а также в прочие форматы, которые определяются пользователем при помощи таблиц стилей XSL.

Самым явным ограничением является то, что оболочка WMIC не имеет встроенную полноценную поддержку и обработку событий WMI. Также нужно отметить, что оболочка является не очень удобной для пользователей, всё из-за того, что разработчики данной оболочки уделили больше внимания ее функциональным особенностям, а не эргономике.

1.5 Причины и цели создания оболочки PowerShell

После предыдущих оболочек разработчики компании Microsoft задумали сделать более мощную среду для работы из командной строки, а также для написания сценариев, так появилась новая оболочка под названием Windows PowerShell. Целей разработки данной оболочки было достаточно много. Основной и наиболее веской из них является то, что существовала необходимость создания такой среды для составления сценариев, которая в большей степени подходила бы для актуальных версий ОС Windows, а также обладала бы повышенной функциональностью и была бы простой в использовании, нежели любые аналогичные разработки создателей других ОС. Еще одной целью является то, что среда должна иметь возможность решить задачи, которые стоят перед системными администраторами, более того также необходимо удовлетворить требования разработчиков ПО путем предоставления им средств для быстрой реализации интерфейсов для управления создаваемыми приложениями.

Язык сценариев PowerShell легок в изучении, написание на нем сценариев, реализующих обращение к внешним объектам, проще, чем на предыдущих разработках Microsoft VBScript и JScript. Вообще, оболочка PowerShell получилась намного эргономичнее, а ее функционал намного мощнее, нежели у предшественников cmd.exe и WSH, однако существуют и ограничения, основной проблемой, которая препятствует распространению инструмента, является то, что оболочка имеет возможность работать не во всех версиях ОС Windows. Поддержка начинается только с Windows XP Service Pack 2 при условии, что установлен пакет .NET Framework 2.0.

Основной отличительной чертой оболочки PowerShell, является то, что в качестве единицы обработки и передачи информации служит не последовательность байтов, а объект, представляющий собой данные с методами. Из-за этого работа в среде PowerShell упрощается за счет отсутствия надобности выполнения действий по выделению нужной информации из символьного потока данных.

1.6 Отличие PowerShell от других оболочек — ориентация на объекты

При разработке любого языка программирования одним из основных является вопрос о том, какие типы данных и каким образом будут в нем представлены. При создании PowerShell разработчики решили не изобретать ничего нового и воспользоваться унифицированной объектной моделью .NET. Данный выбор был сделан по нескольким причинам.

Во-первых, платформа .NET повсеместно используется при разработке программного обеспечения для Windows и предоставляет, в частности, общую информационную схему, с помощью которой разные компоненты операционной системы могут обмениваться данными друг с другом.

Во-вторых, объектная модель .NET является самодокументируемой: каждый объект .NET содержит информацию о своей структуре. При интерактивной работе это очень полезно, так как появляется возможность непосредственно из командной строки выполнить запрос к определенному объекту и увидеть описание его свойств и методов, то есть понять, какие именно манипуляции можно проделать с данным объектом, не изучая дополнительной документации с его описанием.

В-третьих, работая в оболочке с объектами, можно с помощью их свойств и методов легко получать нужные данные, не занимаясь разбором и анализом символьной информации, как это происходит во всех традиционных оболочках командной строки, ориентированных на текст.[1]

2. Описание существующих платформ и технологий виртуализации

2.1 Определение виртуализации

Виртуализация – это программная технология, которая делает возможной одновременное выполнение нескольких ОС и приложений на одном сервере. Виртуализация представляет собой процесс создания не физической, а виртуальной версии какого-либо объекта вычислительной среды. Виртуализация может применяться к компьютерам, операционным системам, устройствам хранения, приложениям или сетям.

При виртуализации программное обеспечение используется для имитации наличия оборудования и создания виртуальной компьютерной системы.

Виртуализация является способом перевода инфраструктуры на динамический уровень согласно модели оптимизации.

С помощью виртуализации создается виртуальный образ операционной системы компьютера, вычислительной сети или накопительного устройства. Виртуальная машина сосуществует «внутри» компьютера с обычной операционной системой. Средства виртуализации помогают превратить один компьютер в более функциональную группу виртуальных машин.

2.2 Виды виртуализации

2.2.1 Аппаратная виртуализация.

Аппаратная виртуализация была реализована в процессорах Intel 386 и носила название V86 mode. Этот режим работы 8086-го процессора позволял запускать параллельно несколько DOS-приложений. В настоящее время аппаратная виртуализация позволяет запускать несколько независимых виртуальных машин в соответствующих разделах аппаратного пространства компьютера. Аппаратная виртуализация является продолжением эволюции уровней абстрагирования программных платформ – от многозадачности до уровня виртуализации.

Аппаратная виртуализация должна поддерживаться на уровне процессора посредством применения соответствующей архитектуры и введения дополнительных инструкций для предоставления прямого доступа к ресурсам процессора из гостевых операционных систем.

2.2.2 Программная виртуализация

При такой виртуализации создается виртуальная операционная система («виртуальная машина») в рамках «реальной» (основной) ОС. При реализации данного подхода все виртуальные машины используют для взаимодействия с аппаратной частью ядро базовой операционной системы, но при этом для каждого пользователя они выглядят, как отдельные независимые операционные системы. Это представлено на рисунке 1.



Рисунок 1 – Структура программной виртуализации

2.2.3 Прочие виды виртуализации

Также существуют еще такие виды виртуализации, как:

- виртуализация серверов - подразумевает запуск на одном физическом сервере нескольких виртуальных серверов;
- виртуализация приложений - технология, которая предназначена для разделения и изоляции приложений на стороне клиента;
- виртуализация рабочих столов - предназначена для изменения управления компьютерными ресурсами конечных пользователей путем виртуализации рабочих столов компьютеров и консолидации их на централизованных серверах;

- виртуализация представлений - подразумевает эмуляцию интерфейса пользователя, пользователь видит приложение и работает с ним на своём компьютере, хотя на самом деле приложение выполняется на удалённом сервере, а пользователю передаётся лишь интерфейс удалённого приложения;
- виртуализация локальных сетей - полное воспроизведение физической сети программным методом;
- виртуализация уровня ОС - основана на поддержании операционной системой хоста нескольких изолированных разделов;
- паравиртуализация - технология, при которой гостевые ОС подготавливаются для исполнения в виртуальной среде, для чего их ядро модифицируется;
- виртуализация ресурсов - разделение одного физического узла на несколько частей, каждая из которых видна для пользователя в качестве отдельного сервера;
- виртуализация систем хранения данных - представляет собой технологию, предоставляющую возможность построения изолированного пространства хранения данных с единым интерфейсом управления на основе ресурсов физических накопителей информации, и обеспечивающую возможность получения необходимой информации по сети;
- полная виртуализация - технология при которой виртуальная машина полностью виртуализует все аппаратное обеспечение при сохранении гостевой ОС в неизменном виде.[2]

2.3 Платформы виртуализации

2.3.1 Microsoft Hyper-V

Microsoft Hyper-V – система аппаратной виртуализации для x64-систем на основе гипервизора.

Технология виртуализации Hyper-V реализована на основе архитектуры гипервизора, который предоставляет службы и ресурсы для создания, управления и выполнения виртуальных машин. Hyper-V формирует среду

виртуального аппаратного обеспечения, виртуальные жесткие диски (virtual hard disks, VHD), а также виртуальные сети, что позволяет выполнять и сохранять виртуальные машины, а также вести обмен между ними.

Гипервизор работает непосредственно над аппаратным обеспечением, позволяет одновременно выполняться внутри разделов нескольким операционным системам с обособленными наборами виртуализированных ресурсов и обеспечивает изоляцию разделов друг от друга (посредством реализации политик доступа для критических системных ресурсов, таких как память и процессоры).

Hyper-V функционирует в несколько потоков и одновременно выполняет одну или более виртуальных машин (каждая машина в собственном потоке выполнения). Каждая виртуальная машина предоставляет (гостевой ОС и приложениям) набор виртуализированных устройств, которые абстрагируют имеющееся аппаратное обеспечение, что обеспечивает переносимость ВМ между различными физическими серверами (работающими под управлением Hyper-V).

Роль Hyper-V в ОС Windows Server реализуется набором компонентов, включающих:

- гипервизор (называемый гипервизором Windows);
- компоненты режима ядра;
- компоненты пользовательского режима.[3]

2.3.2 VMware

Представляет собой серверную виртуализацию первого типа и включает в себя следующие компоненты:

- VMware vSphere — платформа для виртуализации ИТ-инфраструктуры предприятия. vSphere подразумевает одновременное использование ESXi хостов (x86) и vCenter Server для их централизованного управления;
- VMware ESX — гипервизор первого типа, являющийся частью продукта VMware vSphere, требующий отдельной установки;

- VMware ESXi — версия VMware ESX без локальной сервисной консоли;

- VMware Server — программное обеспечение, позволяющее создавать и запускать виртуальные машины, использовать возможности создания снимков, а также удалённо управлять ими;

- vCenter Server — главный инструмент управления для администраторов vSphere. Он предоставляет единую точку контроля за всеми компонентами в виртуальном центре обработки данных и выполняет некоторые важные функции. vCenter позволяет конфигурировать новые ESXi хосты, хранилище, сеть и характеристики виртуального оборудования различных инфраструктурных компонентов, создавать и импортировать новые виртуальные машины, производить мониторинг и отчёты о производительности гостевой ОС, виртуальных машин и ESXi хостов. vCenter управляет правами, разрешениями, ролями на различных уровнях виртуальной инфраструктуры, унифицирует ресурсы различных хостов ESXi и позволяет обеспечивать общий доступ к ним для любых ВМ в центре обработки данных.[4]

2.3.3 Citrix Xen

Xen — это монитор виртуальных машин или гипервизор с поддержкой паравиртуализации для процессоров x86 архитектуры, распространяющийся с открытым исходным кодом. Xen может организовать совместное безопасное исполнение нескольких виртуальных машин на одной физической системе с производительностью близкой к непосредственной.

Xen обеспечивает:

- производительность виртуальных машин близкая к производительности при непосредственном исполнении на железе;

- возможность живой миграции работающих виртуальных машин между хостами;

- поддержку до 32 виртуальных процессоров на одну гостевую машину с возможностью горячего добавления процессоров;
- поддержку платформ x86/32, x86/32 с PAE, x86/64, IA64, а также частичная поддержка платформ ARM и PPC;
- поддержка аппаратной виртуализации для запуска немодифицированных операционных систем;
- поддержка широкого множества оборудования.[5]

2.3.4 KVM

KVM - программное решение, обеспечивающее виртуализацию в среде Linux на платформе x86, которая поддерживает аппаратную виртуализацию на базе Intel VT, либо AMD SVM.

Программное обеспечение KVM состоит из загружаемого модуля ядра, называемого `kvm.ko`, предоставляющего базовый сервис виртуализации, процессорно-специфического загружаемого модуля `kvm-amd.ko`, либо `kvm-intel.ko`, и компонентов пользовательского режима.

KVM позволяет виртуальным машинам использовать немодифицированные образы дисков QEMU, VMware и других, содержащие операционные системы. Каждая виртуальная машина имеет своё собственное виртуальное аппаратное обеспечение: сетевые карты, диск, видеокарту и другие устройства.

Графические утилиты управления:

- Virtual Machine Manager: создание, редактирование, запуск и остановка виртуальных машин, основанных на KVM;
- ConVirt: создание, редактирование, запуск и остановка виртуальных машин, основанных на KVM, как горячую, так и холодную миграцию с других VM хостов;
- Proxmox Virtual Environment - Свободная виртуализация, основанная на KVM и OpenVZ;
- AQEMU: создание, редактирование, запуск, остановка, сохранение

снапшотов виртуальных машин, основанных на KVM и QEMU;

- GKVM: для рабочей среды GNOME. Создание, редактирование, запуск, остановка, сохранение снапшотов виртуальных машин KVM;

- QtEmu: графическая оболочка для QEMU и KVM, написана на Qt4.

Интерфейс на десяти языках.[6]

2.3.5 Платформа, используемая при разработке практикума

При разработке практикума была использована платформа виртуализации Microsoft Hyper-V. В качестве операционных систем использовались Windows Server 2012 R2 в вариантах установки с графическим интерфейсом и без графического интерфейса, а также Windows 8.1 Enterprise Edition. В качестве основного средства управления применяется PowerShell. Такой выбор обусловлен тем, что PowerShell является универсальным средством управления и применяется не только в Hyper-V, но и в VMware vSphere.

3 Описание подготовки виртуальных стендов, используемых при выполнении заданий практикума

3.1 Установка ПО VirtualBox и компонента Extension Pack

В качестве ПО виртуализации используется Oracle VirtualBox 4.3.30. Дистрибутив был взят с официального сайта компании Oracle[7].

Для установки необходимо запустить файл установки VirtualBox.

После запуска установщика на экран будет выведено диалоговое окно с возможностью выбора пути установки и компонентов программы. Программа может включать следующие компоненты:

- VirtualBox Application – основной компонент VirtualBox, без него запуск программы будет невозможен;
- VirtualBox USB Support – компонент поддержки USB драйверов;
- VirtualBox Networking – компонент поддержки драйверов сетевых устройств;
- VirtualBox Python 2.x Support – компонент поддержки Python.

Диалоговое окно выбора пути установки и компонентов программы представлено на рисунке 2.

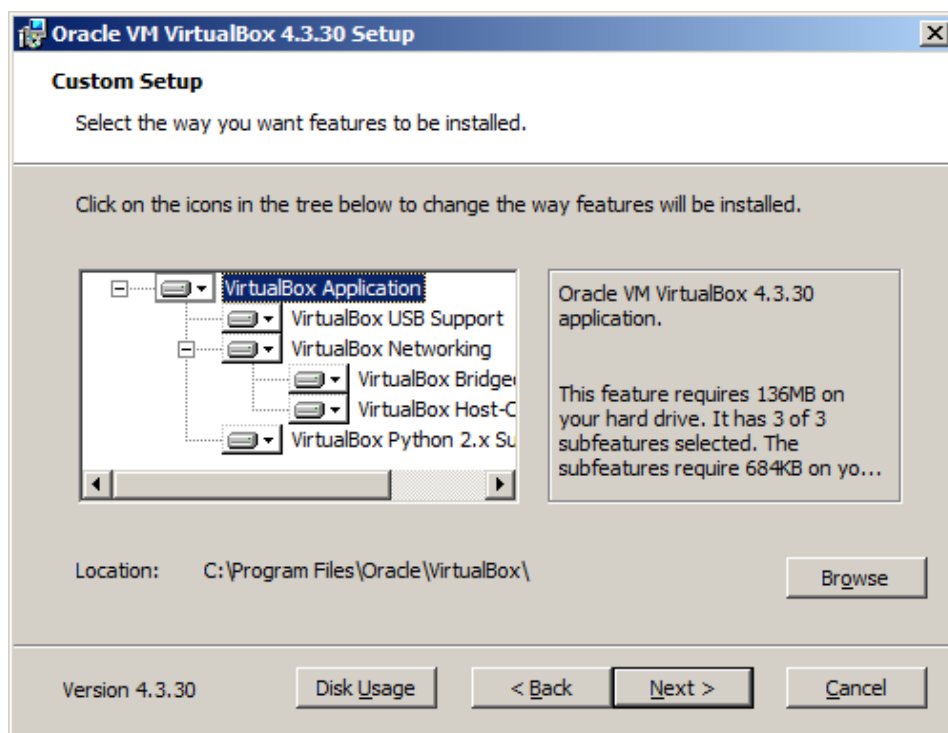


Рисунок 2 – Выбор компонентов установки

В данном диалоговом окне выбираются все компоненты, т.к. они необходимы для нормального функционирования программы. Путь установки остается без изменений.

На следующем шаге имеется возможность выбрать следующие параметры:

- «Create a shortcut on the desktop» – выбор этой опции позволяет после установки автоматически поместить ярлык программы на Рабочий стол;

- «Create a shortcut in the Quick Launch Bar» - выбор этой опции позволяет после установки автоматически поместить ярлык программы в строку быстрого доступа;

- «Register file associations» - выбор этой опции позволяет после установки программы зарегистрировать в системе все ассоциированные с программой типы файлов.

Диалоговое окно выбора параметров представлено на рисунке 3.

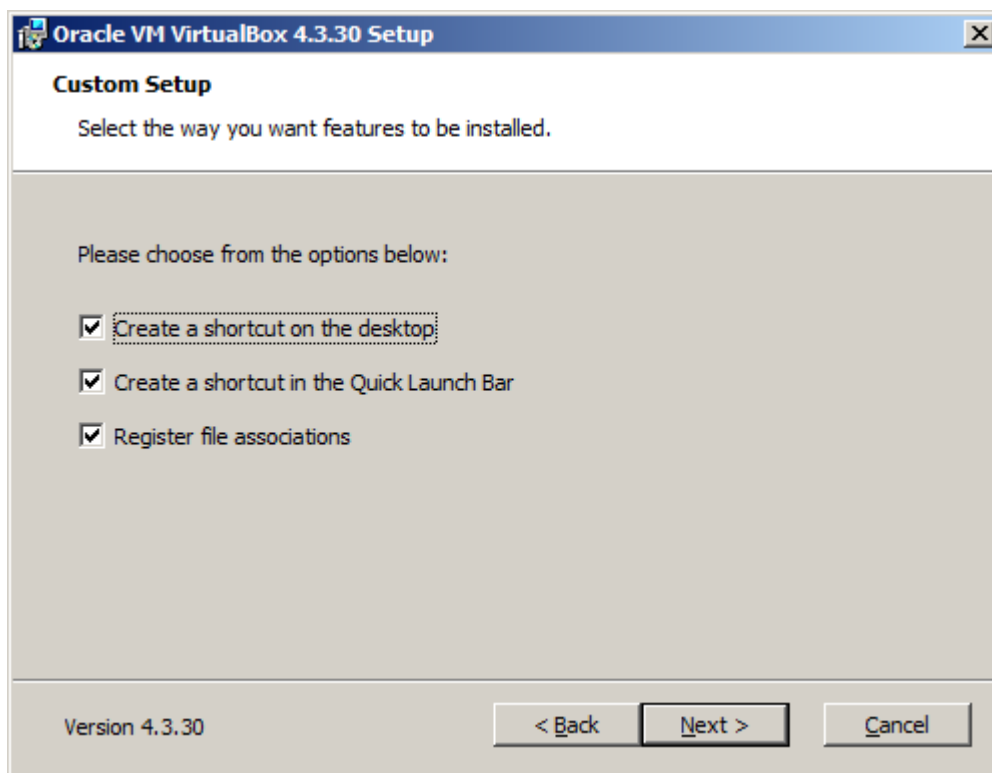


Рисунок 3 – Окно выбора параметров создания ярлыков

Были выбраны опции Create a shortcut on the desktop и Register file associations.

На следующем шаге выводится предупреждение, что установка отключит текущие сетевые соединения. Диалоговое окно с предупреждением показано на рисунке 4.



Рисунок 4 – Окно подтверждения установки

Для подтверждения установки необходимо нажать кнопку «Yes».

После установки VirtualBox необходимо установить пакет расширений (Extension Pack), который обеспечивает возможность использования подсистемы USB, создание общей папки обмена между гостевой и основной ОС. Кроме того пакет расширений обеспечивает корректный запуск ВМ.

Для установки пакета файл установки помещается в папку с установленным ПО VirtualBox, далее происходит запуск этого файла. Установка проходит в автоматическом режиме без вмешательства пользователя.

3.2 Создание виртуальных машин

После установки ПО VirtualBox оно было использовано для создания следующих ВМ:

- одна ВМ с ОС Windows Server 2012 R2 с установленным графическим интерфейсом;
- две ВМ с ОС Windows Server 2012 R2 без установленного графического интерфейса;
- одна ВМ с ОС Windows 8.1 Enterprise Edition.

Для создания ВМ необходимо запустить ПО VirtualBox Manager. Далее в главном окне VirtualBox Manager необходимо нажать кнопку «Создать». На экран выводится диалоговое окно, где необходимо указать имя новой ВМ, а также тип и версию ОС, которая будет использоваться в ВМ. Это показано на рисунке 5.

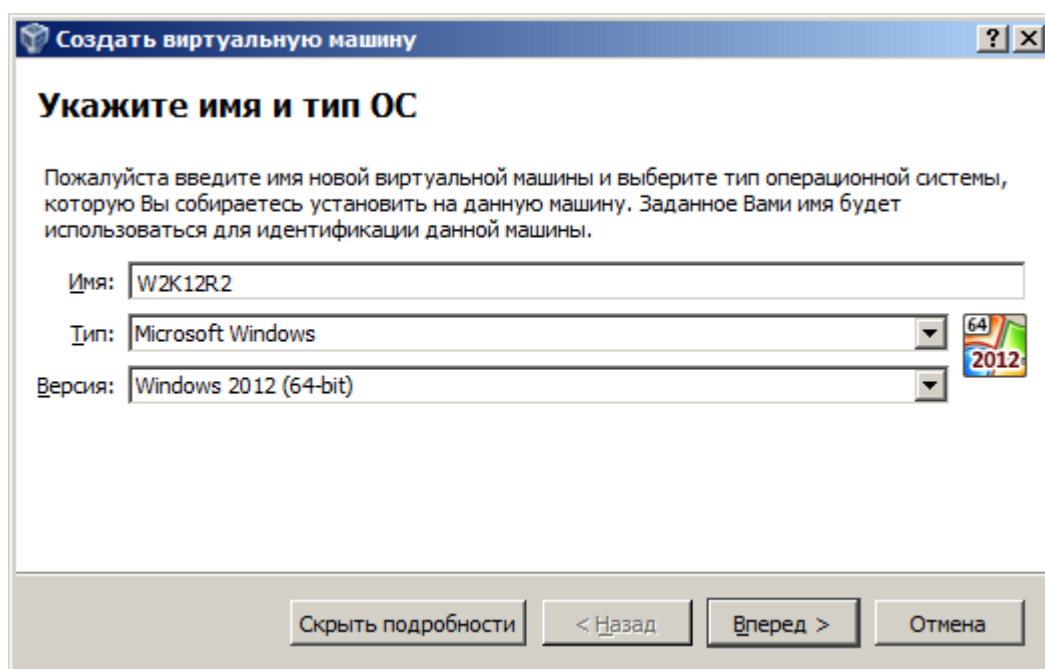


Рисунок 5 – Окно выбора параметров ОС

Первой была создана ВМ с ОС Windows Server 2012 R2 с графическим интерфейсом. В качестве имени используется W2K12R2-SRV, что и обозначает наименование, версию ОС и ее роль в разрабатываемом виртуальном стенде. Тип ОС – Microsoft Windows. Версия ОС – Windows 2012 (64-bit).

На следующем шаге был определен объем оперативной памяти, который будет выделен для ВМ. Рекомендуемый объем зависит от типа и версии ОС. Для ОС Windows Server 2012 R2 рекомендуемый объем – 2048 Мб. Можно выбрать и другой объем ОЗУ, учитывая характеристики ПК, на который будет установлен стэнд. Объем оперативной памяти можно будет изменить и после создания ВМ. Диалоговое окно выбора объема ОЗУ показано на рисунке 6.

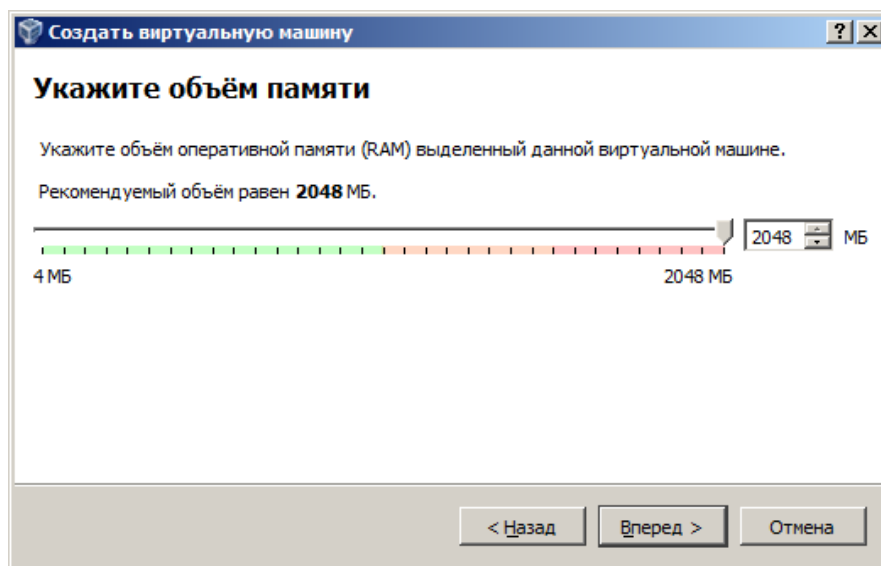


Рисунок 6 – Окно настройки объема оперативной памяти

Объем оперативной памяти остался без изменений, т.к. рекомендуемый мастером установки объем является достаточным для нормального функционирования ВМ с установленной ОС Windows 2012 R2 с графическим интерфейсом.

На следующем шаге была произведена настройка виртуального жесткого диска. Есть три варианта настройки:

- «Не подключать виртуальный жесткий диск»;
- «Создать новый виртуальный жесткий диск»;
- «Использовать существующий жесткий диск» – в этом случае подготовленный образ жесткого диска выбирается вручную.

Был выбран пункт «Создать новый виртуальный жесткий диск». На следующем шаге указывается тип файла, определяющий формат нового диска. Был выбран формат .vdi, это показано на рисунке 7.

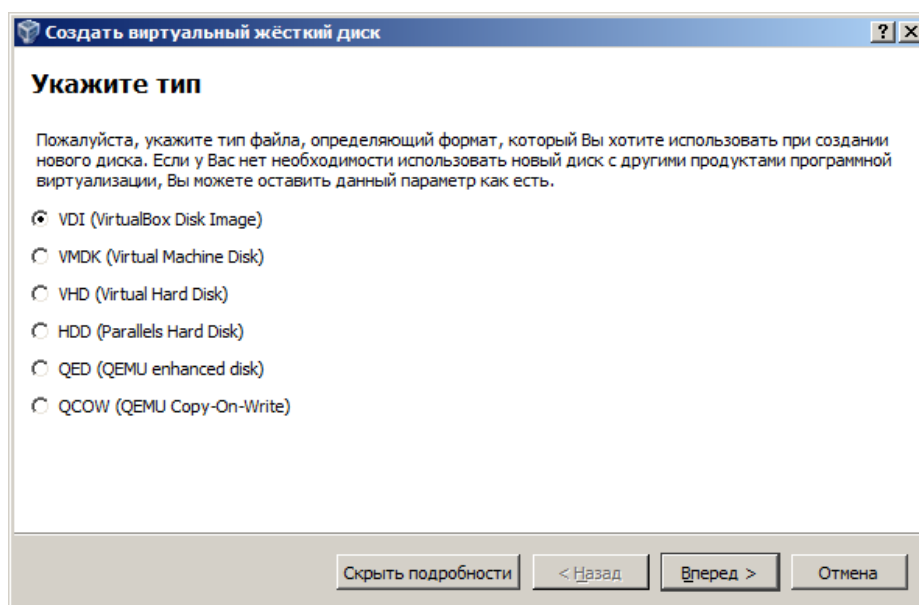


Рисунок 7 – Окно выбора типа файла виртуального жесткого диска

Был выбран формат .vdi, т.к. он обеспечивает переносимость ВМ с одного компьютера на другой.

На следующем шаге был произведен выбор типа жесткого диска :

- динамический виртуальный жесткий диск – использует место реального носителя по мере заполнения виртуального;
- фиксированный виртуальный жесткий диск – использует фиксированное количество места на реальном жестком диске, однако у такого диска повышено быстродействие.

Был выбран тип жесткого диска - динамический виртуальный жесткий диск, т.к. это позволяет экономить место на жестком диске ПК. Экономия осуществляется за счет того, что при создании ВМ файл с жестким диском виртуальной машины имеет минимальный размер. Размер файла увеличивается по мере заполнения жесткого диска на виртуальной машине.

На следующем шаге указывается имя и размер нового виртуального жесткого диска. Имя диска выбирается исходя из названия ВМ, размер диска выбирается равным 40Гб, именно столько необходимо для нормального функционирования создаваемой ВМ.

Таким образом, была создана ВМ со следующим параметрами:

- имя ОС - W2K12R2-SRV;

- тип ОС – Microsoft Windows;
- версия ОС – Windows 2012 (64-bit);
- объем ОЗУ – 2 Гб;
- вариант настройки виртуального жесткого диска – «Создать новый виртуальный жесткий диск»;
- тип жесткого диска – «Динамический виртуальный жесткий диск»;
- объем жесткого диска ВМ – 40 Гб.

Установленный объем файла виртуального жесткого диска является ограничением и определяет максимальный размер файла.

Для создания других ВМ был применен аналогичный порядок действий, за исключением того, что для создания ВМ с ОС Windows Server 2012 R2 без графического интерфейса размер ОЗУ составляет 1 Гб. А для создания ВМ с ОС Windows 8.1 Enterprise выбираются соответствующие тип и версия гостевой ОС.

3.3 Установка гостевых ОС

После создания ВМ с пустыми жесткими дисками необходимо установить гостевые ОС.

Для созданных ВМ были установлены следующие ОС:

- ОС Windows Server 2012 R2 с установленным графическим интерфейсом;
- ОС Windows Server 2012 R2 без установленного графического интерфейса;
- ОС Windows 8.1 Enterprise.

Сначала была установлена ОС Windows Server 2012 R2 с графическим интерфейсом. Процедура установки производилась в соответствии с документацией компании Microsoft. При завершении установки был определен пароль учетной записи администратора.

Аналогичным образом были установлены гостевые ОС Windows Server 2012 R2 без графического интерфейса. Выбор типа ОС производится на одном

из шагов установки. ВМ с установленной ОС без графического интерфейса используется для имитации серверов виртуализации, работающих в таком режиме. ВМ с установленной ОС Windows 8.1 Enterprise используется для имитации рабочей станции локальной сети, на которой работает обычный пользователь.

3.4 Настройка ВМ с помощью VirtualBox Manager

После создания ВМ и установки на них гостевых ОС необходимо настроить виртуальную локальную сеть, а также ряд других параметров с использованием менеджера ВМ.

Для настройки ВМ необходимо вызвать контекстное меню правой кнопкой мыши, далее выбрать пункт «Настроить» в контекстном меню. Будет выведено диалоговое окно настроек.

В категории «Общие» на вкладке «Дополнительно» необходимо настроить параметры буфера обмена:

- параметр «Общий буфер обмена» - Двухнаправленный;
- параметр «Drag'n'Drop» - Двухнаправленный.

Это позволит более удобно пользоваться буфером обмена между ВМ. Результат настройки показан на рисунке 8.

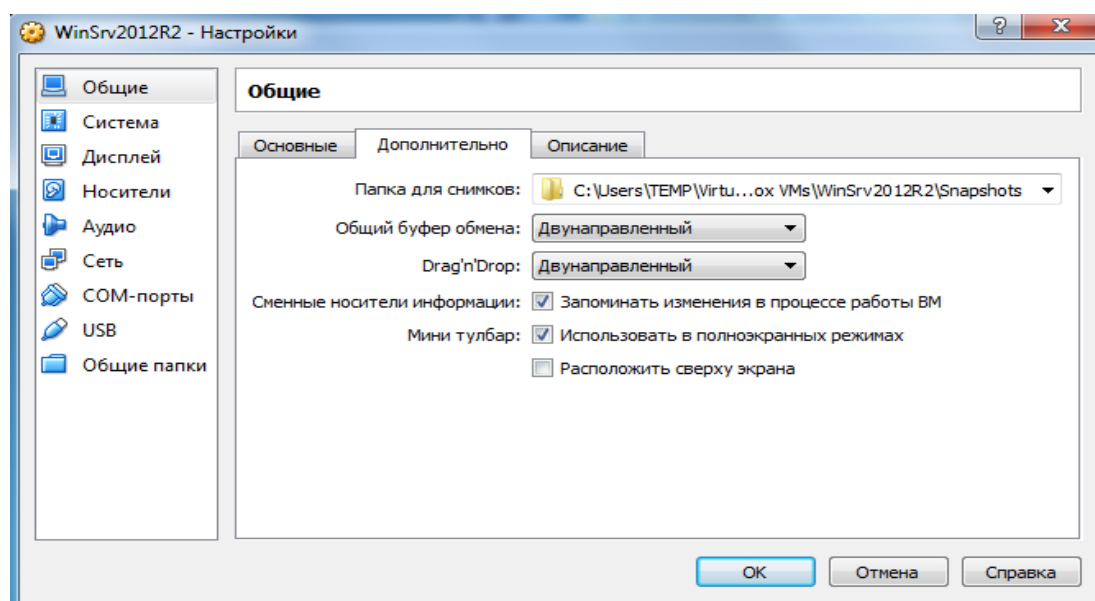


Рисунок 8 – Диалоговое окно настроек буфера обмена

Далее в категории «Сеть» на вкладке «Адаптер 1» необходимо включить сетевой адаптер и выбрать «Тип подключения» внутренняя сеть. Имя сети intnet. После этого, нажав кнопку со стрелкой вниз, открыть меню «Дополнительно». В нем выбирается тип адаптера «Intel PRO/1000 MT Server (82545EM)», для параметра «Неразборчивый режим» надо установить значение «Запретить». Результат настройки параметров локальной сети для созданной ВМ показан на рисунке 9.

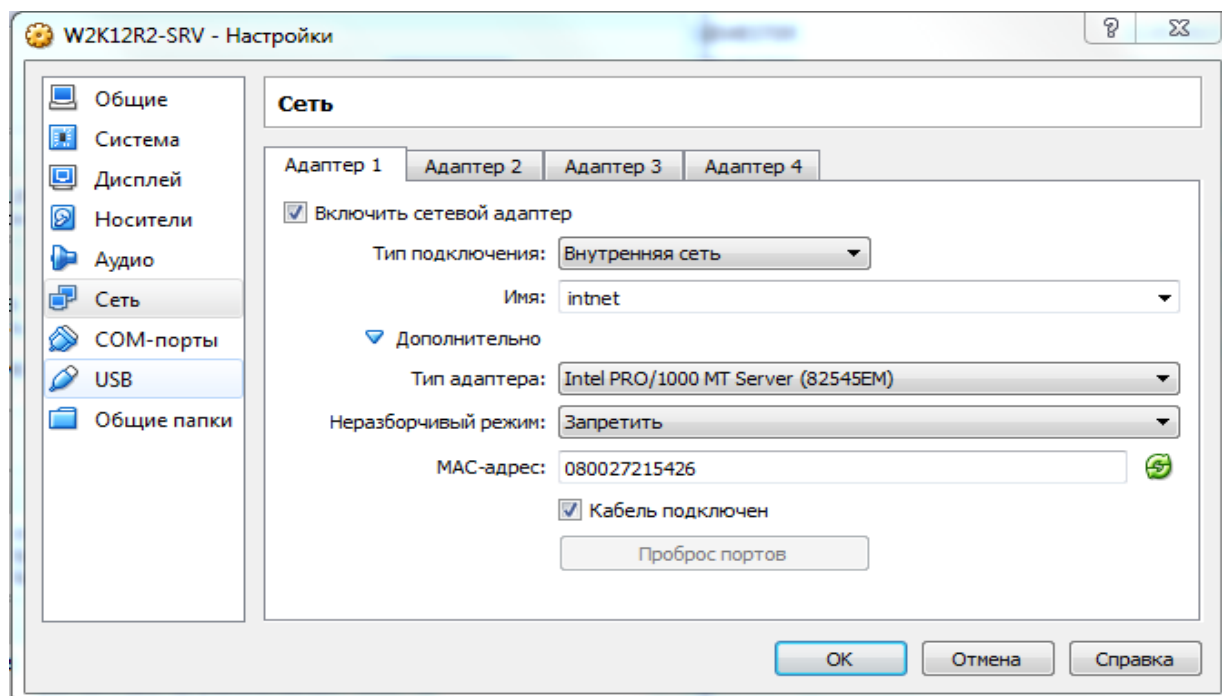


Рисунок 9 – Диалоговое окно настроек сети

Далее настраивается общая папка. Для настройки общей папки необходим установленный пакет расширений Extension Pack. Папка обмена не должна располагаться на системном диске. Чтобы настроить общую папку, в категории «Общие папки» необходимо указать путь общей папки. Результат настройки данного диалогового окна показан на рисунке 10.

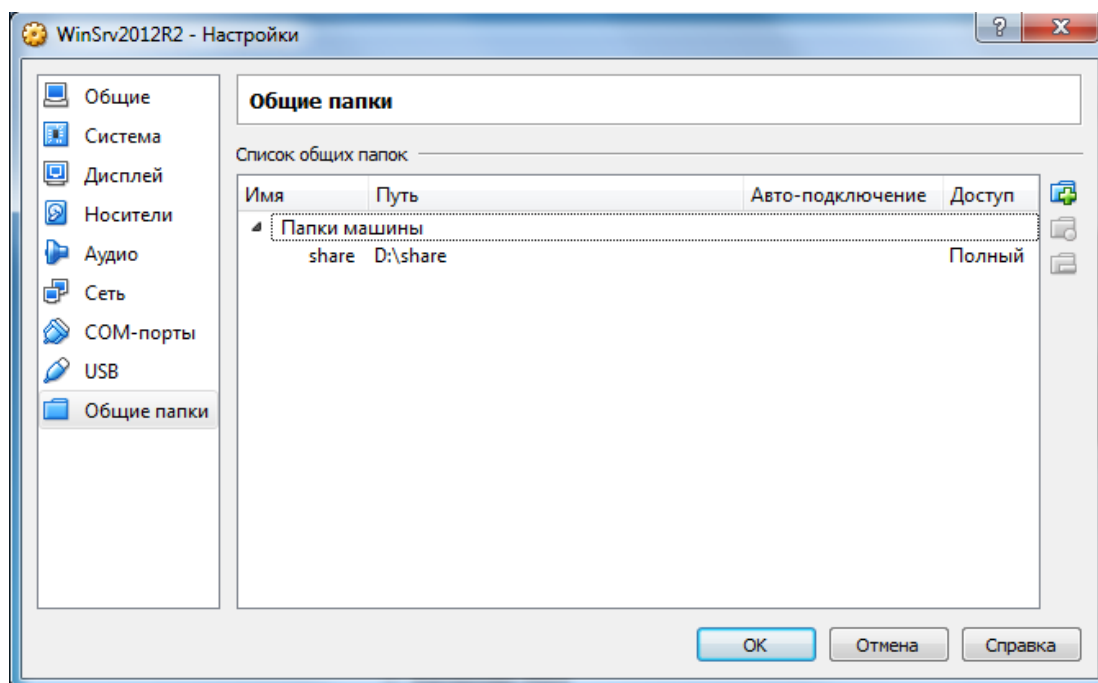


Рисунок 10 – Диалоговое окно настроек общей папки

После настройки параметров для их применения необходимо нажать кнопку «ОК».

3.5 Настройка гостевых ОС

Для настройки сетевых подключений необходимо запустить все ВМ. После этого необходимо осуществить вход под учетной записью администратора на каждой ВМ.

Далее необходимо проверить доступность ВМ в локальной сети intnet. Для этого необходимо запустить командную строку. Далее с помощью команды ping проверяется доступность ВМ. В качестве параметра для команды ping указывается сетевой адрес ВМ, с которой проверяется доступность.

Для того, чтобы уточнить сетевой адрес, необходимо открыть диалоговое окно сетевого адаптера. На ВМ без графического интерфейса для уточнения сетевого адреса используется команда ipconfig с ключом /all

Далее необходимо узнать имя компьютера. После этого имя по умолчанию необходимо сменить:

- для ВМ с ОС Windows Server 2012 R2 с графическим интерфейсом на W2K12R2-SRV;

- для ВМ с ОС Windows Server 2012 R2 без графического интерфейса на W2K12R2-core1 и W2K12R2-core2;

- для ВМ с ОС Windows 8.1 Enterprise на WIN8.1-CL.

Далее необходимо установить компонент ОС Windows, реализующий сервис DNS, поскольку в создающемся виртуальном стенде используется статическая адресация в виду небольшого количества машин.

После установки DNS необходимо создать зону прямого просмотра, в которой будут включены все ВМ. Имя зоны – wps.local.

Для проверки правильности установки DNS необходимо проверить наличие консоли DNS и наличие следующих локальных служб:

- DNS-сервер(только для серверов);
- DNS-клиент.

После проверки в окне свойств сетевого адаптера необходимо отключить динамическое распределение адресов (DHCP) и указать параметры для работы служб DNS.

Далее необходимо к имени компьютера дописать DNS-суффикс wps.local. Это делается для добавления компьютера в зону прямого просмотра.

Для того, чтобы все вышеописанные изменения вступили в силу, необходимо перезагрузить компьютеры.

Для проверки работы DNS необходимо использовать утилиту ping, а в качестве параметров использовать сначала сетевой адрес, а потом DNS-имя. В обоих случаях пакеты должны доходить до пункта назначения.

Далее происходит создание домена и назначение ролей ВМ в локальной сети. Для этого проводится установка служб Active Directory, необходимых для работы домена, а также веб-сервера IIS, сервера сертификатов Active Directory и сервера приложений.

Для каждой роли был установлен набор компонентов, необходимый для ее работы. В роли первичного контроллера домена выступает ВМ W2K12R2-SRV. Именем созданного домена является имя зоны прямого просмотра.

Далее в домен wps.local были добавлены ВМ WIN8.1-CL и W2K12R2-core1.

Для проверки работы домена был выполнен вход под учетной записью доменного администратора.

Далее проверяется синхронизация времени между ВМ домена. Это необходимо для обеспечения корректного управления компьютерами домена с сервера.

После этого настройка гостевых ОС является завершенной.

3.6 Проверка работы оболочки PowerShell

После подготовки гостевых ОС было проведено обновление справки Windows PowerShell. Для этого было настроено подключение ВМ к сети Интернет и загружены необходимые модули.

Была произведена проверка получения справки по отдельным командлетам, для проверки установленного обновления справки.

Была проверена возможность удаленного управления машинами с помощью PowerShell.

Для работы утилиты makecert на ВМ W2K12R2-SRV был установлен Microsoft Windows Platform SDK. Процедура установки производилась в соответствии с документацией компании Microsoft.

После этого настройки Интернет были удалены, поскольку при выполнении заданий практикума использоваться не будут.

3.7 Результат создания виртуального стенда

В результате описанных в подразделах 3.1-3.6 действий, был разработан виртуальный стенд, включающий следующие ВМ:

- одна ВМ с ОС Windows Server 2012 R2 с графическим интерфейсом с именем W2K12R2-SRV;
- две ВМ с ОС Windows Server 2012 R2 без графического интерфейса с именами W2K12R2-core1 и W2K12R2-core2;
- одна ВМ с ОС Windows 8.1 Enterprise с именем WIN8.1-CL.

Каждая ВМ выполняет определенную роль:

- W2K12R2-SRV – первичный контроллер домена, сервер-приложений, центр сертификации, веб-сервер, SMTP-сервер;
- W2K12R2-core1 – член домена, сервер виртуализации;
- W2K12R2-core2 – сервер виртуализации;
- WIN8.1-CL – клиентская машина.

После создания ВМ и объединения их в локальную сеть была выполнена проверка функционирования каждой ВМ отдельно и всей локальной сети, в которую входят созданные ВМ. Результаты проверки показали, что все ВМ и созданный лабораторный стенд функционируют в соответствии с требованиями ТЗ и могут применяться для выполнения заданий практикума.

Состав и структура виртуального стенда показаны в приложении А.

4 Порядок выполнения задания по изучению применения политик выполнения PowerShell для управления запуском сценариев

4.1 Цель задания

Цель задания заключается в разработке сценария PowerShell в соответствии с вариантом и проверку использования этого сценария с применением политики выполнения сценариев. По завершению выполнения задания должен быть разработан отчет по шаблону. Задание включает в себя:

- создание сценария;
- проверка работоспособности сценария при установлении разных значений политики выполнения сценариев при управлении локальным сервером;
- проверка работоспособности сценария при установлении разных значений политики выполнения сценариев при управлении удаленным сервером.

4.2 Создание сценариев с использованием Powershell

При использовании Powershell можно создавать сценарии, которые представляют собой последовательность команд Powershell. Фактически сценарии являются программами, созданными с помощью обложки Powershell. При работе со сценариями в Powershell предусмотрены следующие возможности:

- создание сценариев;
- подписывание сценариев;
- редактирование сценариев;
- создание запланированных заданий.

Сценарий Powershell является обычным текстовым файлом с расширением «.ps1». Поэтому для создания файла сценария в Проводнике можно использовать стандартные возможности ОС Windows по созданию текстовых файлов. Для выполнения практического задания необходимо создать каталог Scripts на диске C:\ виртуальной машины. Перейти в этот каталог и

щелкнуть правой кнопкой мыши. В контекстном меню выбрать команду «Создать» и в выпадающем списке выбрать – «Текстовый документ». Ввести имя файла, присвоить ему имя и расширение ps1.

По умолчанию, из соображений безопасности, при двойном клике он не запускается, а открывается в текстовом редакторе. Необходимо проверить установку параметра «Скрывать расширения у зарегистрированных файлов» в свойствах каталога. Для этого необходимо открыть Панель управления и выбрать раздел Параметры папок. Затем в диалоговом окне Параметры папок перейти на вкладку Вид и выбрать параметр «Скрывать расширения для зарегистрированных типов файлов», пример показан на рисунке 11. Если параметр не установлен, то создание сценариев доступно непосредственно из окна проводника Windows (можно менять расширение на ps1). Если указанный параметр включен, то необходимо менять расширение файла любым доступным методом, например, из командной строки или из файлового менеджера. Для выполнения работы необходимо отключить этот параметр.

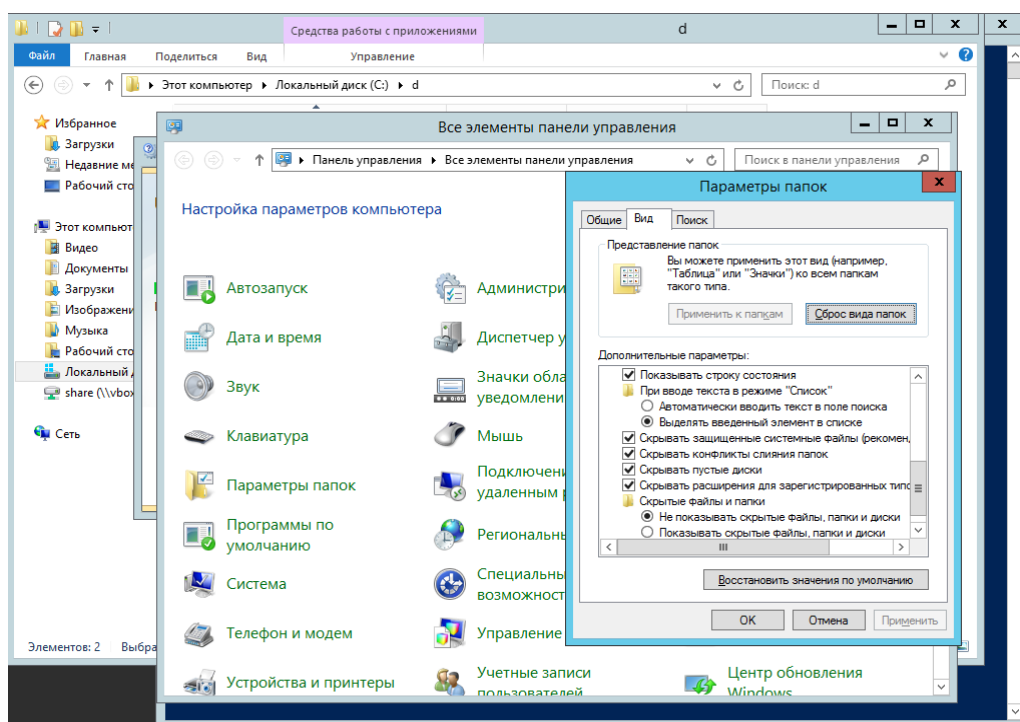


Рисунок 11 – Установка параметра каталогов для изменения расширений

Необходимо на диске C:\ виртуальной машины создать каталог scripts и в этом каталоге создать сценарий script_1.ps1 со следующим содержимым:

Write-Host "Hello!"

Затем запустить оболочку PowerShell, перейти в каталог `c:\scripts` командой

cd c:\scripts

и попробовать запустить сценарий командой

.\script_1.ps1

В результате может быть выдано сообщение об ошибке, показанное на рисунке 12.

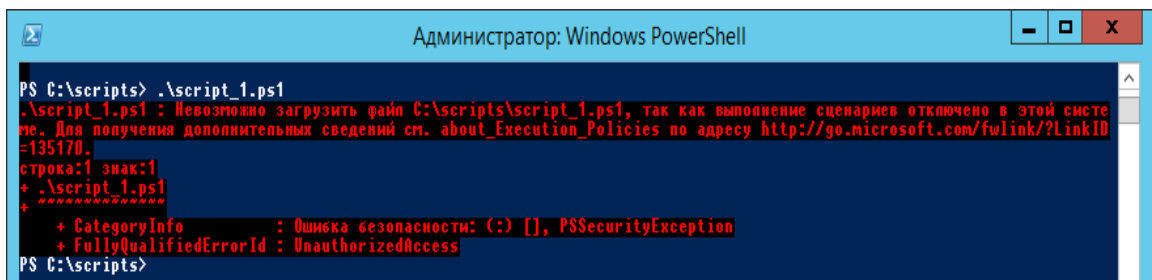


Рисунок 12 – Сообщение о блокировании сценария

Такое сообщение связано с тем, что в настройках по умолчанию, установлена возможность интерактивного выполнения только команд, а выполнение сценариев запрещено.

В случае необходимости уровень выполнения политики (по умолчанию установлен уровень `Restricted`, также может быть установлено значение `RemoteSigned`) можно изменить.

4.3 Настройка политик на локальном сервере

Это процедура применяется для настройки оболочки PowerShell с целью изменения текущего уровня безопасности, который задается параметром «`ExecutionPolicy`», который можно изменить в реестре. Он расположен по следующему пути

[HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\PowerShell\1\ShellIds\Microsoft.PowerShell]

Также можно использовать командлет, который позволяет проверить текущее значение политики выполнения сценариев:

Get-ExecutionPolicy

Существует шесть уровней политики ExecutionPolicy:

- Restricted. Является значением по умолчанию. Означает, что никакие сценарии не могут быть запущены. Единственной возможностью является выполнение одиночных команд;

- AllSigned. Такое значение политики разрешает выполнять сценарии. Сценарий должен быть подписан сертификатом от доверенного источника. Перед выполнением подписанного сценария будет выводиться предупреждение;

- RemoteSigned. Такое значение уровня разрешает выполнять сценарии, но требует, чтобы все сценарии и файлы конфигурации, загруженные из Интернета, были подписаны сертификатом от доверенного источника. Сценариям, выполняющимся на локальном компьютере, подпись не требуется. Предупреждений перед запуском сценариев не выводится;

- Unrestricted. Такая настройка используется чаще всего, но она является потенциально опасной и не рекомендуется к использованию. Такая настройка позволяет запускать неподписанные сценарии, включая загруженные из Интернета;

- Bypass. Ни один сценарий не блокируется. Никаких предупреждений при запуске сценариев не выводится;

- Undefined. Такое значение отменяет действие политики безопасности для сценариев PowerShell.

В случае необходимости уровень разрешений запуска изменяется командой

```
PS C:\scripts> Set-ExecutionPolicy RemoteSigned
```

При этом выдается сообщение, как показано на рисунке 13.

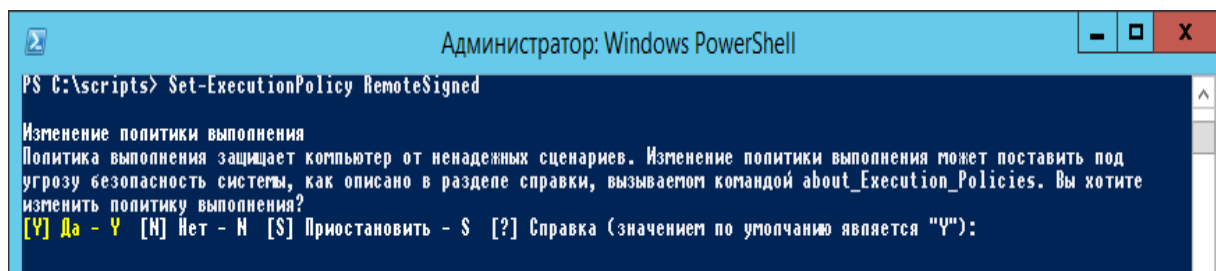


Рисунок 13 – Сообщение при изменении политики выполнения сценариев

Для применения значения, указанного в командлете надо выбрать вариант Y.

После проверки, политике выполнения сценариев устанавливается значение Restricted. При попытке запуска сценария при текущем значении политики выдается следующее сообщение, представленное на рисунке 14.

```
PS C:\scripts> set-executionpolicy restricted

Изменение политики выполнения
Политика выполнения защищает компьютер от ненадежных сценариев. Изменение политики выполнения может поставить под угрозу безопасность системы, как описано в разделе справки, вызываемом командой about_Execution_Policies. Вы хотите изменить политику выполнения?
[Y] Да - Y [N] Нет - N [S] Приостановить - S [?] Справка (значением по умолчанию является "Y"): y
PS C:\scripts> .\script1.ps1
.\script1.ps1 : Невозможно загрузить файл C:\scripts\script1.ps1, так как выполнение сценариев отключено в этой системе. Для получения дополнительных сведений см. about_Execution_Policies по адресу http://go.microsoft.com/fwlink/?LinkID=135170.
строка:1 знак:1
+ .\script1.ps1
+ ~~~~~
+ CategoryInfo          : Ошибка безопасности: (:) [], PSSecurityException
+ FullyQualifiedErrorId : UnauthorizedAccess
```

Рисунок 14 – Запуск сценария при политике Restricted

Далее значение уровня политики устанавливается на AllSigned. Результат запуска сценария при текущем значении политики показан на рисунке 15.

```
PS C:\scripts> set-executionpolicy allsigned

Изменение политики выполнения
Политика выполнения защищает компьютер от ненадежных сценариев. Изменение политики выполнения может поставить под угрозу безопасность системы, как описано в разделе справки, вызываемом командой about_Execution_Policies. Вы хотите изменить политику выполнения?
[Y] Да - Y [N] Нет - N [S] Приостановить - S [?] Справка (значением по умолчанию является "Y"): y
PS C:\scripts> .\script1.ps1
.\script1.ps1 : Не удается загрузить файл C:\scripts\script1.ps1. Файл C:\scripts\script1.ps1 не имеет цифровой подписи. Невозможно выполнить сценарий в указанной системе. Для получения дополнительных сведений о выполнении сценариев и настройке политики выполнения см. about_Execution_Policies по адресу http://go.microsoft.com/fwlink/?LinkID=135170.
строка:1 знак:1
+ .\script1.ps1
+ ~~~~~
+ CategoryInfo          : Ошибка безопасности: (:) [], PSSecurityException
+ FullyQualifiedErrorId : UnauthorizedAccess
```

Рисунок 15 – Запуск сценария при политике AllSigned

Далее значение уровня политики устанавливается на Unrestricted. Результат запуска сценария при текущем значении политики показан на рисунке 16.

```
PS C:\scripts> set-executionpolicy unrestricted

Изменение политики выполнения
Политика выполнения защищает компьютер от ненадежных сценариев. Изменение политики выполнения может поставить под угрозу безопасность системы, как описано в разделе справки, вызываемом командой about_Execution_Policies. Вы хотите изменить политику выполнения?
[Y] Да - Y [N] Нет - N [S] Приостановить - S [?] Справка (значением по умолчанию является "Y"): y
PS C:\scripts> .\script1.ps1
Hello!
```

Рисунок 16 – Запуск сценария при политике Unrestricted

Далее значение уровня политики устанавливается на Bypass. Результат запуска сценария при текущем значении политики показан на рисунке 17.

```
PS C:\scripts> set-executionpolicy bypass
Изменение политики выполнения
Политика выполнения защищает компьютер от ненадежных сценариев. Изменение политики выполнения может поставить под угрозу безопасность системы, как описано в разделе справки, вызываемой командой about_Execution_Policies. Вы хотите изменить политику выполнения?
[Y] Да - Y [N] Нет - N [S] Приостановить - S [?] Справка (значением по умолчанию является "Y"): y
PS C:\scripts> .\script1.ps1
Hello!
```

Рисунок 17 – Запуск сценария при политике Bypass

Далее значение уровня политики устанавливается на Undefined. Результат запуска сценария при текущем значении политики показан на рисунке 18.

```
PS C:\scripts> set-executionpolicy undefined
Изменение политики выполнения
Политика выполнения защищает компьютер от ненадежных сценариев. Изменение политики выполнения может поставить под угрозу безопасность системы, как описано в разделе справки, вызываемой командой about_Execution_Policies. Вы хотите изменить политику выполнения?
[Y] Да - Y [N] Нет - N [S] Приостановить - S [?] Справка (значением по умолчанию является "Y"): y
PS C:\scripts> .\script1.ps1
.\script1.ps1 : Невозможно загрузить файл C:\scripts\script1.ps1, так как выполнение сценариев отключено в этой системе. Для получения дополнительных сведений см. about_Execution_Policies по адресу http://go.microsoft.com/fwlink/?LinkID=135170.
строка:1 знак:1
+ .\script1.ps1
+ ~~~~~
+ CategoryInfo          : Ошибка безопасности: (:) [], PSSecurityException
+ FullyQualifiedErrorId : UnauthorizedAccess
```

Рисунок 18 – Запуск сценария при политике Undefined

Далее значение уровня политики устанавливается на RemoteSigned. Результат запуска сценария при текущем значении политики показан на рисунке 19.

```
PS C:\scripts> set-executionpolicy remotesigned
Изменение политики выполнения
Политика выполнения защищает компьютер от ненадежных сценариев. Изменение политики выполнения может поставить под угрозу безопасность системы, как описано в разделе справки, вызываемой командой about_Execution_Policies. Вы хотите изменить политику выполнения?
[Y] Да - Y [N] Нет - N [S] Приостановить - S [?] Справка (значением по умолчанию является "Y"): y
PS C:\scripts> .\script1.ps1
Hello!
```

Рисунок 19 – Запуск сценария при политике RemoteSigned

4.4 Настройка политик на удаленном сервере

Настройка политик запуска проходит на удаленной машине посредством PowerShell. Запуск PowerShell на сервере без графического интерфейса производится при помощи команды *powershell*, которая вводится непосредственно в командной строке.

После этого на сервере с графическим создается папка общего доступа, в которую помещается созданный ранее сценарий для проверки политик выполнения сценариев.

Далее значение уровня политики устанавливается на Restricted. Результат запуска сценария при текущем значении политики показан на рисунке 20.

```

PS C:\> set-executionpolicy restricted
PS C:\> .\script1.ps1
.\script1.ps1 : Невозможно загрузить файл C:\script1.ps1, так как выполнение сценариев отключено в этой системе. Для получения дополнительных сведений см. about_Execution_Policies по адресу http://go.microsoft.com/fwlink/?LinkID=135170.
строка:1 знак:1
+ .\script1.ps1
+ ~~~~~
+ CategoryInfo          : Ошибка безопасности: (:) [], PSSecurityException
+ FullyQualifiedErrorId : UnauthorizedAccess
PS C:\>

```

Рисунок 20 – Запуск сценария при политике Restricted

Далее значение уровня политики устанавливается на AllSigned. Результат запуска сценария при текущем значении политики показан на рисунке 21.

```

PS C:\> set-executionpolicy allsigned
PS C:\> .\script1.ps1
.\script1.ps1 : Не удастся загрузить файл C:\script1.ps1. Файл C:\script1.ps1 не имеет цифровой подписи. Невозможно выполнить сценарий в указанной системе. Для получения дополнительных сведений о выполнении сценариев и настройке политики выполнения см. about_Execution_Policies по адресу http://go.microsoft.com/fwlink/?LinkID=135170.
строка:1 знак:1
+ .\script1.ps1
+ ~~~~~
+ CategoryInfo          : Ошибка безопасности: (:) [], PSSecurityException
+ FullyQualifiedErrorId : UnauthorizedAccess
PS C:\>

```

Рисунок 21 – Запуск сценария при политике AllSigned

Далее значение уровня политики устанавливается на Unrestricted. Результат запуска сценария при текущем значении политики показан на рисунке 22.

```

PS C:\> set-executionpolicy unrestricted
PS C:\> .\script1.ps1
hello!

```

Рисунок 22 – Запуск сценария при политике Unrestricted

Далее значение уровня политики устанавливается на Bypass. Результат запуска сценария при текущем значении политики показан на рисунке 23.

```

PS C:\> set-executionpolicy bypass
PS C:\> .\script1.ps1
hello!

```

Рисунок 23 – Запуск сценария при политике Bypass

Далее значение уровня политики устанавливается на Undefined. Результат запуска сценария при текущем значении политики показан на рисунке 24.

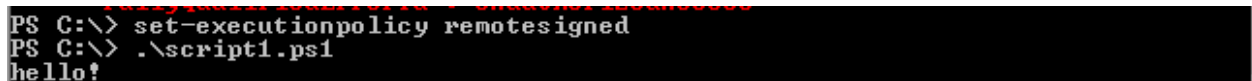
```

PS C:\> set-executionpolicy undefined
PS C:\> .\script1.ps1
.\script1.ps1 : Невозможно загрузить файл C:\script1.ps1, так как выполнение сценариев отключено в этой системе. Для получения дополнительных сведений см. about_Execution_Policies по адресу http://go.microsoft.com/fwlink/?LinkID=135170.
строка:1 знак:1
+ .\script1.ps1
+ ~~~~~
+ CategoryInfo          : Ошибка безопасности: (:) [], PSSecurityException
+ FullyQualifiedErrorId : UnauthorizedAccess
PS C:\>

```

Рисунок 24 – Запуск сценария при политике Undefined

Далее значение уровня политики устанавливается на RemoteSigned. Результат запуска сценария при текущем значении политики показан на рисунке 25.



```
PS C:\> set-executionpolicy remotesigned
PS C:\> .\script1.ps1
hello!
```

Рисунок 25 – Запуск сценария при политике RemoteSigned

При выполнении задания необходимо разработать сценарий PowerShell в соответствии с вариантом. Затем этот сценарий необходимо запускать при различных значениях политики выполнения сценариев. Результаты выполнения сценариев необходимо представлять в виде снимков экрана для формирования отчета о выполнении задания.

5 Порядок выполнения задания по изучению использования сертификатов для управления запуском сценариев PowerShell

5.1 Цель задания

Цель задания заключается в разработке сценария PowerShell в соответствии с вариантом, создании сертификата с помощью утилиты `makecert`, создания сертификата с помощью средств встроенных в PowerShell 4.0, а также подписании сценариев с помощью созданных сертификатов. После подписывания сценариев проверяется работоспособность политик выполнения сценариев для подписанных сценариев. По завершению выполнения задания должен быть разработан отчет по шаблону. Задание включает в себя:

- создание сценария;
- подписание сценария с помощью `makecert`;
- проверка работоспособности подписанного сценария при установлении разных значений политики выполнения сценариев при управлении локальным сервером;
- проверка работоспособности подписанного сценария при установлении разных значений политики выполнения сценариев при управлении удаленным сервером.
- подписание сценария с помощью штатных средств PowerShell;
- проверка работоспособности подписанного сценария при установлении разных значений политики выполнения сценариев при управлении локальным сервером;
- проверка работоспособности подписанного сценария при установлении разных значений политики выполнения сценариев при управлении удаленным сервером.

5.2 Подписывание сценариев с помощью утилиты `makecert`

Для формирования сертификата с целью подписания сценариев PowerShell может быть использована программа `makecert.exe`, которая входит в состав Microsoft .NET Framework SDK или Microsoft Windows Platform SDK. на

данный момент этот вариант не рекомендуется компанией Microsoft в связи с тем, что существуют командлеты непосредственно в оболочке PowerShell, используемые при работе с сертификатами, однако makecert.exe не является запрещенным.

Второй механизм подписания сценариев – командлет в PowerShell New-SelfSignedCertificate. Данный командлет был введен в PowerShell 4.0 и старше.

При выполнении практического задания будут использоваться оба варианта создания сертификатов и подписания сценариев.

Сначала происходит процесс формирования сертификата

Для начала работы с makecert.exe необходимо установить пакет Microsoft Windows Platform SDK для использования этой программы. После установки для запуска открыть командную строку и ввести команду

```
cd "C:\Program Files\Microsoft SDKs\Windows\v7.1\Bin"
```

Для создания сертификатов необходимо ввести команду

```
makecert -n "CN=PowerShell Local Certificate Root" -a sha1 -eku 1.3.6.1.5.5.7.3.3 -r -sv root.pvk root.cer -ss Root -sr localMachine
```

После ввода этой команды будут выданы запросы на ввод пароля для файлов ключей. В качестве паролей можно использовать пароль администратора.

Следующая команда формирования сертификата

```
makecert -pe -n "CN=PowerShell User" -ss MY -a sha1 -eku 1.3.6.1.5.5.7.3.3 -iv root.pvk -ic root.cer
```

Также будет запрошен пароль для подписи. Необходимо ввести пароль администратора.

В результате должны быть созданы два файла root.cer и root.pvk. Для проверки успешно ли прошло создание сертификатов необходимо использовать команду в оболочке PowerShell

```
PS C:> Get-ChildItem cert:\CurrentUser\My -codesign
```

Будет выдано сообщение, как показано на рисунке 26.

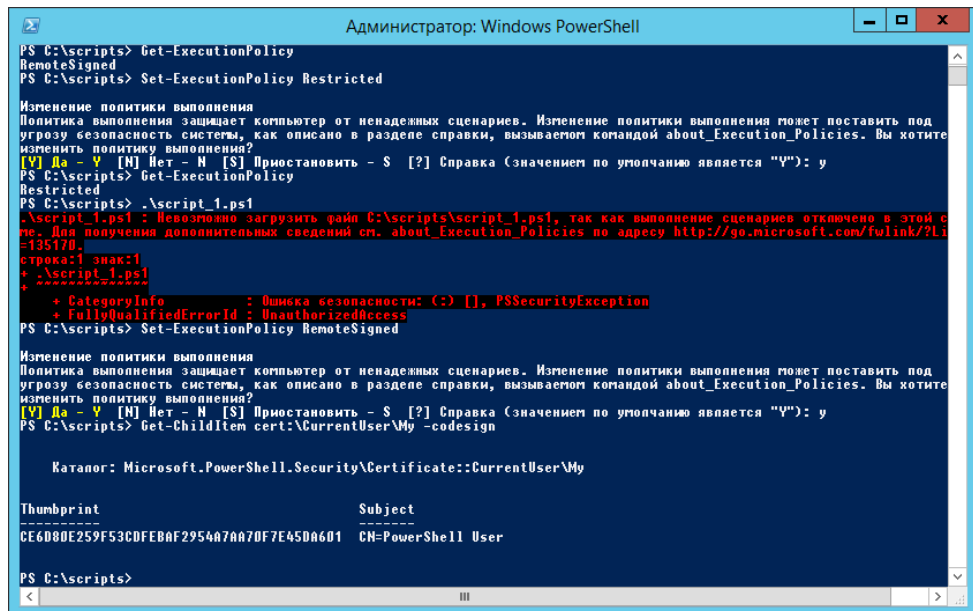


Рисунок 26 – Результат проверки сертификатов

Далее полученный сертификат необходимо установить в системе. Для этого нужно его найти на жестком диске. Как правило, созданные сертификаты находятся в папке с файлом makecert.exe и имеют расширение *.cer. Установка производится двойным щелчком по файлу сертификата и дальнейшим нажатием кнопки «Установить сертификат...». Запускается мастер импорта сертификатов, в нем производится выбор расположения хранилища и расположение сертификата в хранилище. Расположением хранилища выбирается текущий пользователь, а расположение сертификата в хранилище выбирается автоматически мастером на основе типа сертификата. После этого сертификат можно использовать для подписания сценариев.

Для подписания можно использовать пример, содержащийся в справочном руководстве (get-help about_signing), представляющий собой готовый фрагмент кода сценария:

```

## add-signature.ps1

## Signs a file

param([string] $file=$(throw "Please specify a filename."))

$cert = @(Get-ChildItem cert:\CurrentUser\My -codesigning)[0]

Set-AuthenticodeSignature $file $cert

```

Для использования этого файла необходимо, в соответствии с указаниями, приведенными в примере, выполнить следующие команды PowerShell:

```
$cert = @(Get-ChildItem cert:\CurrentUser\My -codesigning)[0]
```

Затем следует выполнить команду

```
Set-AuthenticodeSignature $file $cert
```

В качестве переменной *\$file* необходимо указать имя подписываемого сценария. Вначале указать сценарий *add-signature.ps1*. После подписания его можно будет использовать в дальнейшем. После выполнения этой команды будет выдано сообщение, как показано на рисунке 27.

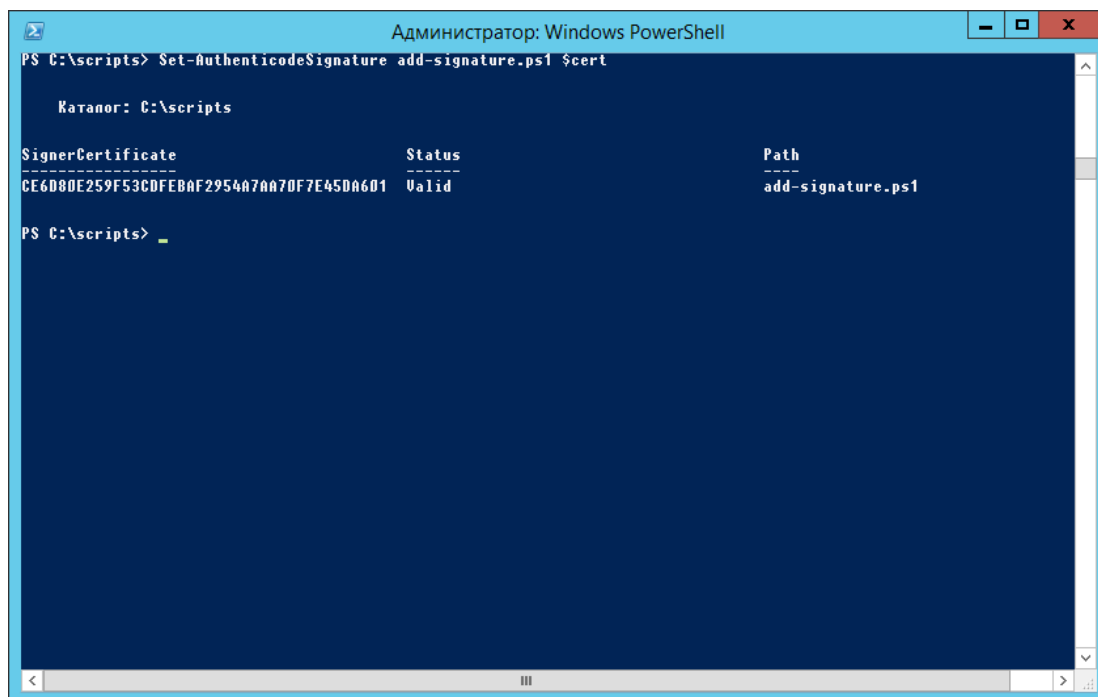


Рисунок 27 – Результат подписания сценария *add-signature.ps1*

Затем аналогично подписывается сценарий *script_1.ps1*.

Если при подписывании сценария возвращается ошибка `UnknownError`, то необходимо сохранить сценарий в кодировке UTF-8. После подписания двух сценариев необходимо проверить действие политики. Для этого установить значение политики безопасности для проверки подписи сценариев `AllSigned`. Затем запустить сценарий *script_1.ps1*. Будет выдан запрос о доверии издателю (см. рисунок 28).

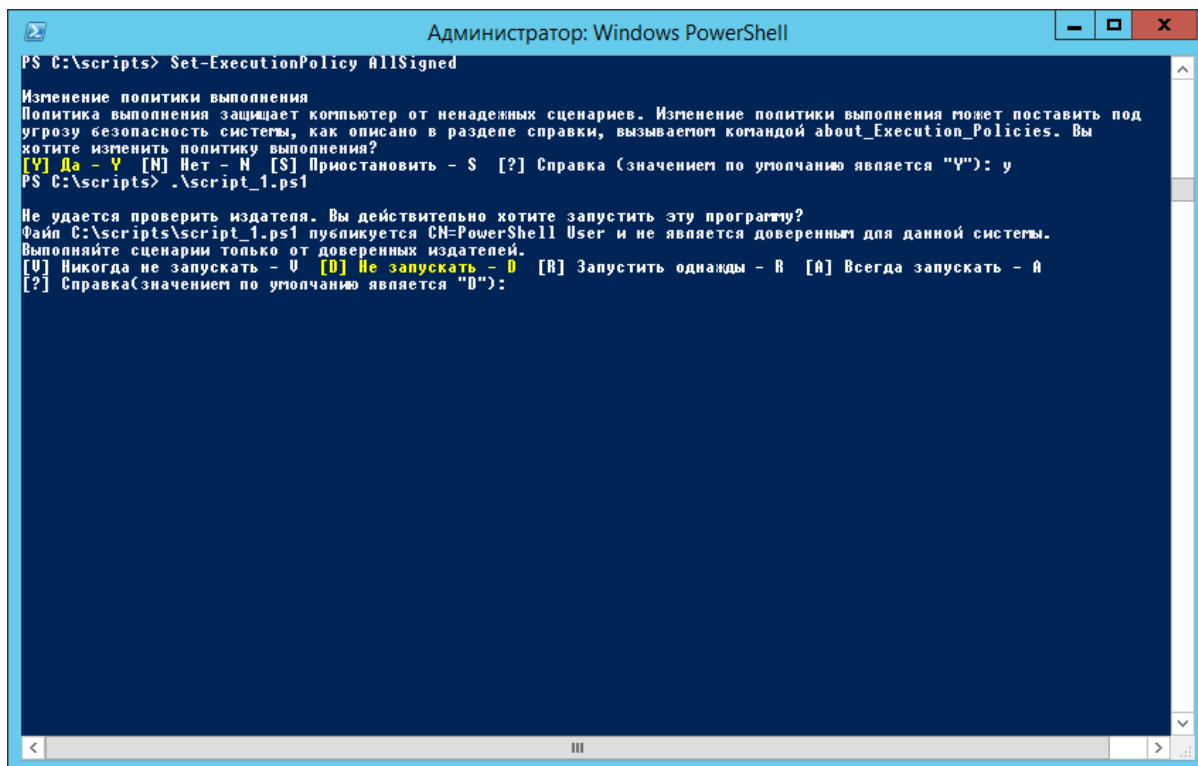


Рисунок 28 – Запрос для подтверждения запуска сценария

В качестве варианта выбрать вариант R – запустить однажды. После этого сценарий будет выполнен. Если будет выбран вариант A – запускать всегда, сценарий будет запускаться без запроса подтверждения подписи.

Если этот подписанный сценарий выложить в открытый доступ, то при включённой политике AllSigned, после его запуска будет выдано предупреждение о том, что подпись сертификата не может быть проверена и сценарий не будет запущен. Это произойдет из-за того, что для удаленного компьютера этот сценарий подписан неизвестным издателем, для которого доверие не установлено. Он может быть запущен только после подписания тем пользователем, который планирует запускать этот сценарий.

В случае если уже есть развернутый доменный Центр Сертификации, то можно выполнить следующую последовательность действий:

- получить доменный сертификат для подписывания (Code Sign);
- выполнить следующую последовательность команд:

```
$cert = @(dir cert:\CurrentUser\My -codesigning)[0]
```

```
Set-AuthenticodeSignature $file $cert,
```

где *\$file* – путь к сценарию и *\$cert* – объект сертификата.

После подписания сертификата проверяется работоспособность всех политик выполнения сценариев на локальном сервере.

Значение уровня политики устанавливается на Restricted. Результат запуска сценария при текущем значении политики показан на рисунке 29.

```
PS C:\scripts> set-executionpolicy restricted

Изменение политики выполнения
Политика выполнения защищает компьютер от ненадежных сценариев. Изменение политики выполнения может поставить под угрозу безопасность системы, как описано в разделе справки, вызываемом командой about_Execution_Policies. Вы хотите изменить политику выполнения?
[Y] Да - Y [N] Нет - N [S] Приостановить - S [?] Справка (значением по умолчанию является "Y"): y
PS C:\scripts> .\script1.ps1
.\script1.ps1 : Невозможно загрузить файл C:\scripts\script1.ps1, так как выполнение сценариев отключено в этой системе. Для получения дополнительных сведений см. about_Execution_Policies по адресу http://go.microsoft.com/fwlink/?LinkID=135170.
строка:1 знак:1
+ .\script1.ps1
+ ~~~~~
+ CategoryInfo          : Ошибка безопасности: (:) [], PSSecurityException
+ FullyQualifiedErrorId : UnauthorizedAccess
```

Рисунок 29 – Запуск сценария при политике Restricted

Далее значение уровня политики устанавливается на AllSigned. Результат запуска сценария при текущем значении политики показан на рисунке 30.

```
PS C:\scripts> set-executionpolicy allsigned

Изменение политики выполнения
Политика выполнения защищает компьютер от ненадежных сценариев. Изменение политики выполнения может поставить под угрозу безопасность системы, как описано в разделе справки, вызываемом командой about_Execution_Policies. Вы хотите изменить политику выполнения?
[Y] Да - Y [N] Нет - N [S] Приостановить - S [?] Справка (значением по умолчанию является "Y"): y
PS C:\scripts> .\script1.ps1

Не удастся проверить издателя. Вы действительно хотите запустить эту программу?
Файл C:\scripts\script1.ps1 публикуется CN=PowerShell User и не является доверенным для данной системы. Выполняйте сценарии только от доверенных издателей.
[U] Никогда не запускать - U [D] Не запускать - D [R] Запустить однажды - R [A] Всегда запускать - A [?] Справка (значением по умолчанию является "D"):r
Hello!
```

Рисунок 30 – Запуск сценария при политике AllSigned

Далее значение уровня политики устанавливается на Unrestricted. Результат запуска сценария при текущем значении политики показан на рисунке 31.

```
PS C:\scripts> set-executionpolicy unrestricted

Изменение политики выполнения
Политика выполнения защищает компьютер от ненадежных сценариев. Изменение политики выполнения может поставить под угрозу безопасность системы, как описано в разделе справки, вызываемом командой about_Execution_Policies. Вы хотите изменить политику выполнения?
[Y] Да - Y [N] Нет - N [S] Приостановить - S [?] Справка (значением по умолчанию является "Y"): y
PS C:\scripts> .\script1.ps1
Hello!
```

Рисунок 31 – Запуск сценария при политике Unrestricted

Далее значение уровня политики устанавливается на Bypass. Результат запуска сценария при текущем значении политики показан на рисунке 32.

```

PS C:\scripts> set-executionpolicy bypass

Изменение политики выполнения
Политика выполнения защищает компьютер от ненадежных сценариев. Изменение политики выполнения может поставить под угрозу безопасность системы, как описано в разделе справки, вызываемом командой about_Execution_Policies. Вы хотите изменить политику выполнения?
[Y] Да - Y [N] Нет - N [S] Приостановить - S [?] Справка (значением по умолчанию является "Y"): y
PS C:\scripts> .\script1.ps1
Hello!

```

Рисунок 32 – Запуск сценария при политике Bypass

Далее значение уровня политики устанавливается на Undefined. Результат запуска сценария при текущем значении политики показан на рисунке 33.

```

PS C:\scripts> set-executionpolicy undefined

Изменение политики выполнения
Политика выполнения защищает компьютер от ненадежных сценариев. Изменение политики выполнения может поставить под угрозу безопасность системы, как описано в разделе справки, вызываемом командой about_Execution_Policies. Вы хотите изменить политику выполнения?
[Y] Да - Y [N] Нет - N [S] Приостановить - S [?] Справка (значением по умолчанию является "Y"): y
PS C:\scripts> .\script1.ps1
.\script1.ps1 : Невозможно загрузить файл C:\scripts\script1.ps1, так как выполнение сценариев отключено в этой системе. Для получения дополнительных сведений см. about_Execution_Policies по адресу http://go.microsoft.com/fwlink/?LinkID=135170.
строка:1 знак:1
+ .\script1.ps1
+ ~~~~~
+ CategoryInfo          : Ошибка безопасности: (:) [], PSSecurityException
+ FullyQualifiedErrorId : UnauthorizedAccess

```

Рисунок 33 – Запуск сценария при политике Undefined

Далее значение уровня политики устанавливается на RemoteSigned. Результат запуска сценария при текущем значении политики показан на рисунке 34.

```

PS C:\scripts> set-executionpolicy remotesigned

Изменение политики выполнения
Политика выполнения защищает компьютер от ненадежных сценариев. Изменение политики выполнения может поставить под угрозу безопасность системы, как описано в разделе справки, вызываемом командой about_Execution_Policies. Вы хотите изменить политику выполнения?
[Y] Да - Y [N] Нет - N [S] Приостановить - S [?] Справка (значением по умолчанию является "Y"): y
PS C:\scripts> .\script1.ps1
Hello!

```

Рисунок 34 – Запуск сценария при политике RemoteSigned

При проверке политик запуска сценариев на удаленном сервере значение уровня политики устанавливается только в RemoteSigned, т.к. результат выполнения прочих политик будет таким же, как и в проверке на локальном сервере. Результат выполнения сценария при установленной политике RemoteSigned показан на рисунке 35.

```

PS C:\> set-executionpolicy remotesigned
PS C:\> .\script1.ps1
hello!

```

Рисунок 35 – Результат выполнения сценария при установленной политике RemoteSigned

5.3 Создание сертификатов с помощью встроенных командлетов PowerShell

Для ОС Windows, начиная с версии 8, и Windows Server, начиная с версии 2012 R2, предусмотрена возможность создания сертификатов без использования посторонних утилит, таких как makecert.

Для этого в PowerShell был добавлен модуль PKI (Public Key Infrastructure). Описание командлетов модуля представлено на рисунке 36.

```
PS C:\Program Files\Microsoft SDKs\Windows\v7.1\Bin> get-command -module pki
```

CommandType	Name	ModuleName
Cmdlet	Add-CertificateEnrollmentPolicyServer	pki
Cmdlet	Export-Certificate	pki
Cmdlet	Export-PfxCertificate	pki
Cmdlet	Get-Certificate	pki
Cmdlet	Get-CertificateAutoEnrollmentPolicy	pki
Cmdlet	Get-CertificateEnrollmentPolicyServer	pki
Cmdlet	Get-CertificateNotificationTask	pki
Cmdlet	Get-PfxData	pki
Cmdlet	Import-Certificate	pki
Cmdlet	Import-PfxCertificate	pki
Cmdlet	New-CertificateNotificationTask	pki
Cmdlet	New-SelfSignedCertificate	pki
Cmdlet	Remove-CertificateEnrollmentPolicyServer	pki
Cmdlet	Remove-CertificateNotificationTask	pki
Cmdlet	Set-CertificateAutoEnrollmentPolicy	pki
Cmdlet	Switch-Certificate	pki
Cmdlet	Test-Certificate	pki

Рисунок 36 — Описание модуля PKI

Для создания сертификата используется командлет *New-SelfSignedCertificate*. Описание командлета показано на рисунке 37.

```
PS C:\Program Files\Microsoft SDKs\Windows\v7.1\Bin> get-help new-selfsignedcertificate
```

ИМЯ	New-SelfSignedCertificate
ОПИСАНИЕ	Creates a new self-signed certificate for testing purposes.
СИНТАКСИС	New-SelfSignedCertificate [-CertStoreLocation <String>] [-CloneCert <Certificate>] [-DnsName <String>] [-Confirm] [-WhatIf] [<CommonParameters>]
ОПИСАНИЕ	<p>The New-SelfSignedCertificate cmdlet creates a self-signed certificate for testing purposes. Using the CloneCert parameter, a test certificate can be created based on an existing certificate with all settings copied from the original certificate except for the public key. A new key of the same algorithm and length will be created.</p> <p>If an existing certificate is not being cloned, then an SSL server certificate with the following default settings is created:</p> <ul style="list-style-type: none">-- Subject: Empty-- Key: RSA 2048-- EKUs: Client Authentication and Server Authentication-- Key Usage: Digital Signature, Key Encipherment (aD)-- Validity Period: One year <p>Delegation may be required when using this cmdlet with Windows PowerShell remoting and changing user configuration.</p>

Рисунок 37 — Описание командлета New-SelfSignedCertificate

Командлет *New-SelfSignedCertificate* должен иметь два параметра - *DnsName* (DNS имя компьютера) и *-CertStoreLocation* (место хранения сертификата). В качестве *DnsName* выбирается DNS имя доменного компьютера W2K12R2-SRV, а в качестве *CertStoreLocation* – *cert:\LocalMachine\My*.

Таким образом полная команда имеет следующий вид:

New-SelfSignedCertificate -DnsName test.contoso.com -CertStoreLocation cert:\LocalMachine\My

При этом данная команда создает сертификат и помещает его в персональное хранилище компьютера. Результат выполнения данной команды показан на рисунке 38.

```
PS C:\Program Files\Microsoft SDKs\Windows\v7.1\Bin> New-SelfSignedCertificate -DnsName W1K12R2-SRV -CertStoreLocation cert:\LocalMachine\My

Katanor: Microsoft.PowerShell.Security\Certificate::LocalMachine\My

Thumbprint                                     Subject
-----
32AC28CC85D7EC46CF4329E8E21E214893F83EA6     CN=W1K12R2-SRV
```

Рисунок 38 — Результат выполнения команды

Для проверки создания сертификата используется оснастка *certlm.msc*. В ней в разделе *Personal* хранилища сертификатов компьютера должен появиться новый сертификат. Это показано на рисунке 39.

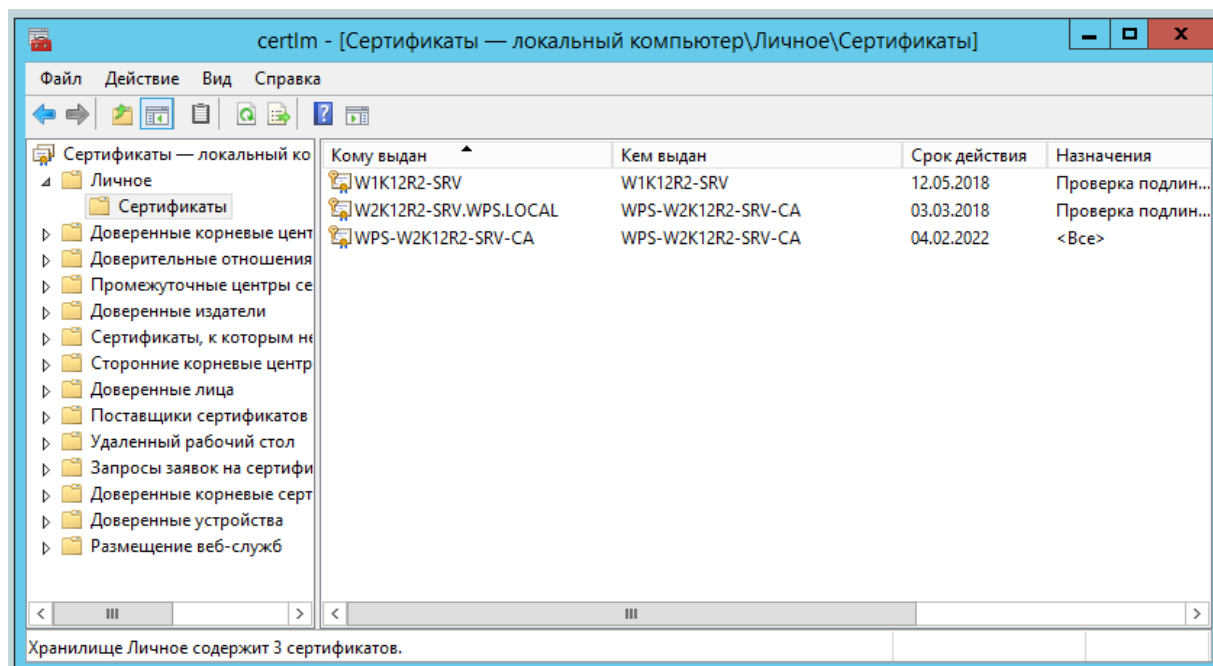


Рисунок 39 — Проверка создания сертификата

Для подписания используется пример сценария из прошлого пункта:

```
## add-sign.ps1
## Signs a file
param([string] $file=$(throw "Please specify a filename."))
$cert = @(Get-ChildItem cert:\LocalMachine\My -codesigning)[0]
Set-AuthenticodeSignature $file $cert
```

Для использования этого файла необходимо, в соответствии с указаниями, приведенными в примере, выполнить следующие команды PowerShell:

```
$cert = @(Get-ChildItem cert:\LocalMachine\My -codesigning)[0]
```

Затем необходимо выполнить команду

```
Set-AuthenticodeSignature $file $cert
```

В качестве переменной \$file необходимо указать имя подписываемого сценария. Сначала необходимо указать сценарий add-sign.ps1. После подписания его можно будет использовать в дальнейшем. После выполнения этой команды на экран будет выведено сообщение, как показано на рисунке 40.

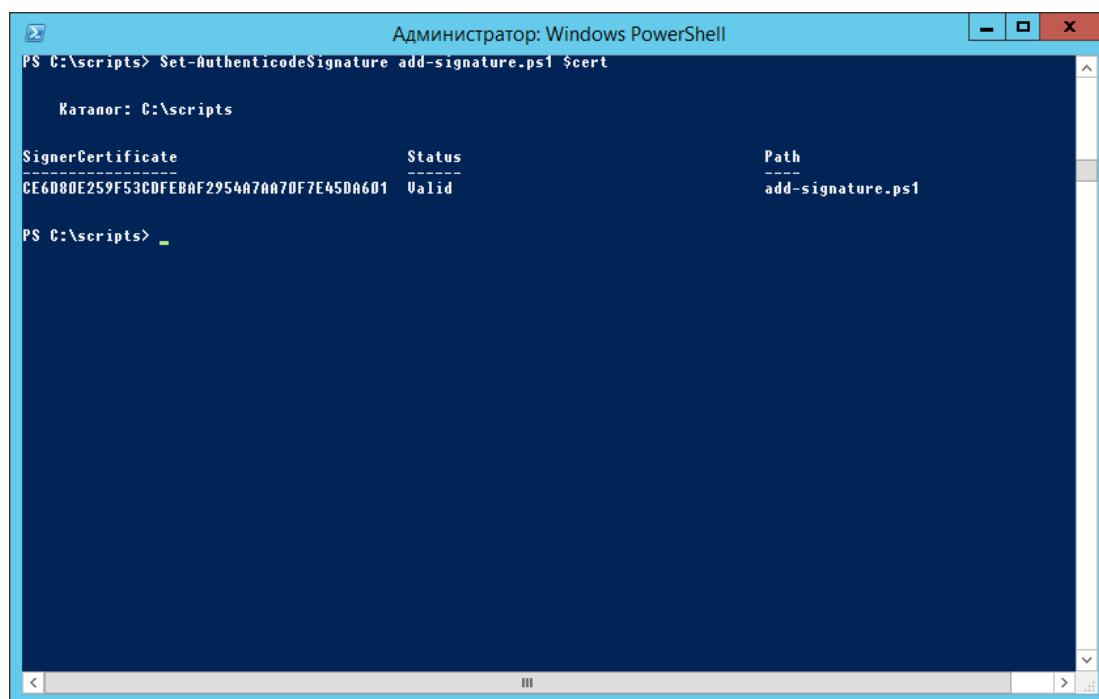
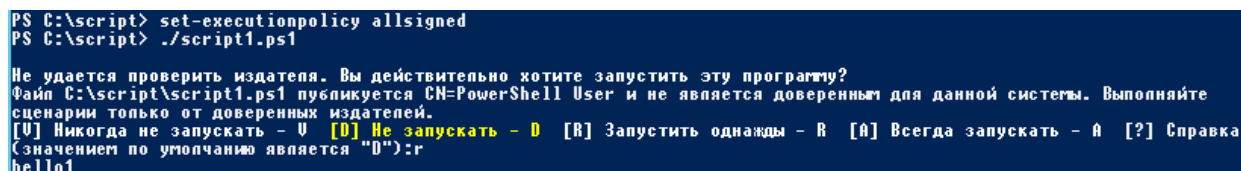


Рисунок 40 – Результат подписания сценария add-sign.ps1

Затем аналогично подписывается сценарий script_1.ps1.

Если при подписывании сценария возвращается ошибка `UnknownError`, то необходимо сохранить сценарий в кодировке UTF-8. После подписания двух сценариев необходимо проверить действие политики. Для этого необходимо установить значение политики безопасности для проверки подписи сценариев `AllSigned`. Затем необходимо запустить сценарий `script1.ps1`. Будет выдан запрос о доверии издателю, показанный на рисунке 41.



```
PS C:\script> set-executionpolicy allsigned
PS C:\script> ./script1.ps1

Не удается проверить издателя. Вы действительно хотите запустить эту программу?
Файл C:\script\script1.ps1 публикуется CN=PowerShell User и не является доверенным для данной системы. Выполняйте
сценарии только от доверенных издателей.
[U] Никогда не запускать - U [D] Не запускать - D [R] Запустить однажды - R [A] Всегда запускать - A [?] Справка
(значение по умолчанию является "D"):r
hello!
```

Рисунок 41 – Запрос для подтверждения запуска сценария

В качестве варианта необходимо выбрать вариант `R` – запустить однажды. После этого сценарий будет выполнен. Если будет выбран вариант `A` – запускать всегда, то сценарий будет запускаться без запроса подтверждения подписи.

Если этот подписанный сценарий выложить в открытый доступ, то при включённой политике `AllSigned`, после его запуска будет выдано предупреждение о том, что подпись сертификата не может быть проверена и сценарий не будет запущен. Это произойдет из-за того, что для удаленного компьютера этот сценарий подписан неизвестным издателем, для которого доверие не установлено. Он может быть запущен только после подписания тем пользователем, который планирует запускать этот сценарий.

В случае если уже есть развернутый доменный Центр Сертификации, то можно выполнить следующую последовательность действий:

- получить доменный сертификат для подписывания (Code Sign);
- выполнить следующую последовательность команд:

```
$cert = @(dir cert:\CurrentUser\My -codesigning)[0]
Set-AuthenticodeSignature $file $cert
```

где `$file` – путь к сценарию и `$cert` – объект сертификата.

После подписания сертификата проверяется работоспособность всех политик выполнения сценариев на локальном сервере. Результат проверки

сценария, подписанного с помощью штатных средств PowerShell, идентичен результату проверки сценария, подписанному с помощью `makecert`.

Второе задание практикума включает разработку сценария PowerShell в соответствии с вариантом, создание сертификата с помощью утилиты `makecert`, подписание сценария с использованием созданного сертификата и проверку работы подписанного сценария при различных значениях политики выполнения сценариев. Также задание включает создание сертификата с помощью средств, встроенных в PowerShell 4.0, подписание сценария с помощью такого сертификата и проверку работоспособности сценария при различных значениях политики выполнения сценариев для подписанных сценариев.

6 Описание порядка выполнения заданий практикума и результатов выполнения заданий

6.1 Порядок выполнения заданий практикума

Выполнение задания практикума предполагает следующую последовательность действий:

- изучение теоретического материала, приведенного в методических указаниях и на лекциях;
- выполнение задания практикума;
- подготовка отчета о выполнении задания практикума;
- защита отчета о выполнении заданий практикума, включающая ответы на контрольные вопросы.

Каждое задание практикума предполагает его выполнение за два академических часа.

Выполнение задания разбито на несколько этапов, выполнение задания включает промежуточный контроль отдельных этапов.

Промежуточный контроль включает проверку выполнения задания преподавателем, при этом проверяется полнота и правильность, а также соответствие результатов пунктам задания.

Порядок выполнения является общим для выполнения заданий практикума.

После выполнения задания должен быть составлен отчет

6.2 Порядок выполнения задания по изучению применения политик выполнения PowerShell для управления запуском сценариев

6.2.1 Начало выполнения задания практикума, подготовка сценария

6.2.1.1 Для начала выполнения задания требуется войти под учетной записью администратора на ВМ с названием W2K12R2-SRV.

6.2.1.2 Для возможности смены расширения текстового файла происходит проверка правильности установки параметров отображения расширений в системе

6.2.1.3 Создается текстовый файл с необходимым тестовым кодом.

6.2.1.4 Текстовому файлу изменяется расширение на *.ps1 для дальнейшего его использования в качестве сценария PowerShell

6.2.2 Проверка настроек политики выполнения сценариев на локальном сервере

6.2.2.1 Запуск оболочки PowerShell, попытка запуска сценария. Фиксация результатов выполнения сценария.

6.2.2.2 Запрос текущего значения политики выполнения сценариев.

6.2.2.3 Установка значения политики выполнения сценариев Restricted. Запуск сценария. Фиксация результатов выполнения сценария.

6.2.2.4 Установка значения политики выполнения сценариев AllSigned. Запуск сценария. Фиксация результатов выполнения сценария.

6.2.2.5 Установка значения политики выполнения сценариев Unrestricted. Запуск сценария. Фиксация результатов выполнения сценария.

6.2.2.6 Установка значения политики выполнения сценариев Bypass. Запуск сценария. Фиксация результатов выполнения сценария.

6.2.2.7 Установка значения политики выполнения сценариев Undefined. Запуск сценария. Фиксация результатов выполнения сценария.

6.2.2.8 Установка значения политики выполнения сценариев RemoteSigned. Запуск сценария. Фиксация результатов выполнения сценария.

6.2.3 Проверка настроек политики на удаленном сервере

6.2.3.1 Для начала проверки настроек политики необходимо войти под учетной записью администратора на VM W2K12R2-core.

6.2.3.2 Запуск оболочки PowerShell, проверка текущего значения политики.

6.2.3.3 Установка значения политики выполнения сценариев Restricted. Запуск сценария. Фиксация результатов выполнения сценария.

6.2.3.4 Установка значения политики выполнения сценариев AllSigned. Запуск сценария. Фиксация результатов выполнения сценария.

6.2.3.5 Установка значения политики выполнения сценариев Unrestricted.
Запуск сценария. Фиксация результатов выполнения сценария.

6.2.3.6 Установка значения политики выполнения сценариев Bypass.
Запуск сценария. Фиксация результатов выполнения сценария.

6.2.3.7 Установка значения политики выполнения сценариев Undefined.
Запуск сценария. Фиксация результатов выполнения сценария.

6.2.3.8 Установка значения политики выполнения сценариев RemoteSigned. Запуск сценария. Фиксация результатов выполнения сценария.

6.2.4 Описание порядка выполнения задания в виде UML-диаграммы

Для графического описания порядка выполнения задания по изучению применения политик выполнения PowerShell для управления запуском сценариев была разработана UML-диаграмма, которая представлена в приложении Б.

6.3 Порядок выполнения задания по изучению использования сертификатов для управления запуском сценариев PowerShell

6.3.1 Начало выполнения задания практикума, подготовка сценария

6.3.1.1 Для начала выполнения задания требуется войти под учетной записью администратора на ВМ с названием W2K12R2-SRV.

6.3.1.2 Запуск оболочки PowerShell

6.3.2 Подписывание сценариев с помощью утилиты makecert

6.3.2.1 Перейти в командную строку из PowerShell командой cmd

6.3.2.2 Создать сертификат с помощью ввода необходимой команды в командной строке. Результат выполнения команды зафиксировать.

6.3.2.3 Сформировать сертификат с помощью ввода необходимой команды в командной строке. Результат выполнения команды зафиксировать.

6.3.2.4 Перейти в оболочку PowerShell

6.3.2.5 Проверить наличие файлов root.pvk и root.cer с помощью ввода определенной команды в PowerShell. Также проверить наличие этих двух

файлов в папке с утилитой makescert.

6.3.2.6 Установить сертификат в системе с помощью мастера импорта сертификатов.

6.3.2.7 Создать файл-сценарий для подписывания сценариев. Поместить в него код, необходимый для подписания сценариев.

6.3.2.8 Подписать файл-сценарий для подписывания сценариев определенной командой. Подписать сценарий из предыдущего задания практикума аналогичным образом.

6.3.3 Проверка настроек политики подписанного сценария на локальном сервере

6.3.3.1 Запрос текущего значения политики выполнения сценариев.

6.3.3.2 Установка значения политики выполнения сценариев Restricted. Запуск сценария. Фиксация результатов выполнения сценария.

6.3.3.3 Установка значения политики выполнения сценариев AllSigned. Запуск сценария. Фиксация результатов выполнения сценария.

6.3.3.4 Установка значения политики выполнения сценариев Unrestricted. Запуск сценария. Фиксация результатов выполнения сценария.

6.3.3.5 Установка значения политики выполнения сценариев Bypass. Запуск сценария. Фиксация результатов выполнения сценария.

6.3.3.6 Установка значения политики выполнения сценариев Undefined. Запуск сценария. Фиксация результатов выполнения сценария.

6.3.3.7 Установка значения политики выполнения сценариев RemoteSigned. Запуск сценария. Фиксация результатов выполнения сценария.

6.3.4 Проверка настроек политики на удаленном сервере

6.3.4.1 Для начала проверки настроек политики необходимо войти под учетной записью администратора на VM W2K12R2-core.

6.3.4.2 Запуск оболочки PowerShell, проверка текущего значения политики.

6.3.4.3 Установка значения политики выполнения сценариев

RemoteSigned. Запуск сценария. Фиксация результатов выполнения сценария.

6.3.5 Подписывание сценариев с помощью штатных командлетов оболочки Windows PowerShell.

6.3.5.1 Создание сертификата с помощью командлета New-SelfSignedCertificate

6.3.5.2 Проверка создания сертификата с помощью оснастки certlm.msc. Фиксация результатов снимком экрана

6.3.5.3 Подписать файл-сценарий для подписывания сценариев определенной командой. Подписать сценарий из предыдущего задания практикума аналогичным образом

6.3.5.4 Проверка работоспособности политик выполнения сценариев с помощью сценария, подписанного сертификатом, созданным с помощью штатных средств оболочки Windows PowerShell

6.3.6 Описание порядка выполнения задания в виде UML-диаграммы

Для графического описания порядка выполнения задания по изучению использования сертификатов для управления запуском сценариев PowerShell была разработана UML-диаграмма, которая представлена в приложении В

7 Описание структуры и содержания отчета о выполнении практических заданий

7.1 Общий перечень данных, содержащихся в отчете

Результаты выполнения заданий, входящих в практикум, оформляются в виде отчетов. Содержание отчета выполнения каждого задания включает в себя:

- создание снимков экранов;
- создание файлов-сценариев с выполняемыми командами;
- показ выполнения пунктов задания и отчетов.

Снимки экранов необходимо делать в процессе выполнения задания практикума и сохранять их. Эти снимки экранов не приведены в рекомендациях по выполнению заданий практикума, поэтому студенты заранее не знают, что на них будет.

Файл-сценарий создается в начале выполнения задания практикума. Далее, в процессе выполнения заданий, команды, примененные в работе, фиксируются в этом файле.

Графически структура отчета о выполнении заданий практикума показана в приложении Г

7.2 Структура отчета по выполнению задания по изучению применения политик выполнения PowerShell для управления запуском сценариев

Отчет должен включать в себя:

- цель задания, представляющую собой краткое описание задач на задание практикума;
- задание практикума, представляющее собой краткий обзор последовательности действий при выполнении задания практикума;
- описание порядка выполнения действий, а именно:
 - описание подготовительных действий для начала выполнения задания;
 - описание проверки настроек политики на локальном сервере;

- описание проверки настроек политики на удаленном сервере;
- вывод, включающий в себя описание навыков, которые были получены в процессе выполнения задания практикума.

Общая структура отчета о выполнении заданий, а также содержание контрольного экземпляра отчета приведены в приложении Д.

7.3 Структура отчета по выполнению задания по изучению использования сертификатов для управления запуском сценариев PowerShell

Отчет должен включать в себя:

- цель задания, представляющую собой краткое описание задач на задание практикума;
- задание практикума, представляющее собой краткий обзор последовательности действий при выполнении задания практикума;
- описание порядка выполнения действий, а именно:
 - описание порядка подписывания сценариев с помощью makecert;
 - описание проверки настроек политики выполнения сценариев на локальном сервере;
 - описание проверки настроек политики на удаленном сервере;
 - описание процесса подписывания сценариев с помощью штатных средств оболочки PowerShell;
- вывод, включающий в себя описание навыков, которые были получены в процессе выполнения задания практикума.

Общая структура отчета о выполнении заданий, а также содержание контрольного экземпляра отчета приведены в приложении Е.

8 Определение порядка контроля выполнения практического задания

8.1 Общий порядок контроля

Порядок контроля выполнения заданий практикума является общим для двух заданий, входящих в практикум, и включает в себя проверку:

- полноты выполнения заданий практикума;
- соответствия созданных снимков экрана пунктам задания;
- функционирования сценариев PowerShell, разработанных при выполнении задания практикума.

После выполнения задания практикума результаты работы предоставляются для проверки. Предъявляются снимки экранов с указанием того пункта задания, при выполнении которого они формировались. Предъявляется текст сценария, сформированного в процессе выполнения работы, для проверки его работоспособности.

После проверки правильности результатов должен быть оформлен отчёт о выполнении задания. К отчету должны прилагаться сформированные файлы-сценарии. Перед защитой работы проверяется правильность оформления отчета и соответствие команд в файле-сценарии выполненному заданию. Если команды соответствуют заданию и отчёт выполнен по требованиям, то он допускается к защите, если не выполнено хотя бы одно из условий, то отчет отправляется на доработку.

Порядок проверки выполнения заданий практикума представлен в приложении Ж.

8.2 Порядок контроля задания по изучению применения политик выполнения PowerShell для управления запуском сценариев

Проверка политик выполнения сценариев на локальном сервере

Должны быть показаны в виде снимков экрана:

- результат проверки сценария для политики Restricted;
- результат проверки сценария для политики Unrestricted;
- результат проверки сценария для политики Undefined;

- результат проверки сценария для политики Bypass;
- результат проверки сценария для политики AllSigned;
- результат проверки сценария для политики RemoteSigned.

Должен быть сформирован файл-сценарий, содержащий:

- команду первичной проверки параметра политики выполнения сценариев;

- команду изменения политики выполнения сценариев на каждый из параметров:

- Restricted;
- Unrestricted;
- Undefined;
- Bypass;
- AllSigned;
- RemoteSigned.

Проверка политик выполнения сценариев на удаленном сервере.

Должны быть показаны в виде снимков экрана:

- результат проверки сценария для политики Restricted;
- результат проверки сценария для политики Unrestricted;
- результат проверки сценария для политики Undefined;
- результат проверки сценария для политики Bypass;
- результат проверки сценария для политики AllSigned;
- результат проверки сценария для политики RemoteSigned.

Должен быть сформирован файл-сценарий, содержащий:

- команду изменения политики выполнения сценариев на каждый из параметров:

- Restricted;
- Unrestricted;
- Undefined;
- Bypass;

- AllSigned;
- RemoteSigned.

8.3 Порядок контроля задания по изучению использования сертификатов для управления запуском сценариев PowerShell

Подписывание сценариев с помощью makecert.

Должны быть показаны в виде снимков экрана:

- результат создания сертификата;
- результат формирования сертификата;
- результаты проверки наличия созданного сертификата в папке с makecert и с помощью командной строки;
- результат подписания сценариев сертификатом.

Должен быть сформирован файл-сценарий, содержащий:

- команду для создания сертификата;
- команду для формирования сертификата;
- команду для проверки наличия созданного сертификата;
- команды для подписания сценариев сертификатом.

Проверка политик выполнения для подписанного сценария на локальном сервере.

Должны быть показаны в виде снимков экрана:

- результат проверки подписанного сценария для политики Restricted;
- результат проверки подписанного сценария для политики Unrestricted;
- результат проверки подписанного сценария для политики Undefined;
- результат проверки подписанного сценария для политики Bypass;
- результат проверки подписанного сценария для политики AllSigned;
- результат проверки подписанного сценария для политики RemoteSigned.

Должен быть сформирован файл-сценарий, содержащий:

- команду первичной проверки параметра политики выполнения сценариев;

– команду изменения политики выполнения сценариев на каждый из параметров:

- Restricted;
- Unrestricted;
- Undefined;
- Bypass;
- AllSigned;
- RemoteSigned.

Проверка политик выполнения сценариев на удаленном сервере.

Должны быть показаны в виде снимков экрана:

- результат проверки подписанного сценария для политики Restricted;
- результат проверки подписанного сценария для политики Unrestricted;
- результат проверки подписанного сценария для политики Undefined;
- результат проверки подписанного сценария для политики Bypass;
- результат проверки подписанного сценария для политики AllSigned;
- результат проверки подписанного сценария для политики

RemoteSigned.

Должен быть сформирован файл-сценарий, содержащий:

– команду изменения политики выполнения сценариев на каждый из параметров:

- Restricted;
- Unrestricted;
- Undefined;
- Bypass;
- AllSigned;
- RemoteSigned.

Подписывание сценариев с помощью штатных средств оболочки Windows PowerShell.

Должны быть показаны в виде снимков экрана:

- результат создания сертификата;
- результаты проверки наличия созданного сертификата в папке с makecert и с помощью командной строки;

- результат подписания сценариев сертификатом.

Должен быть сформирован файл-сценарий, содержащий:

- команду для создания сертификата;
- команду для проверки наличия созданного сертификата;
- команды для подписания сценариев сертификатом.

Проверка политик выполнения для подписанного сценария на локальном сервере.

Должны быть показаны в виде снимков экрана:

- результат проверки подписанного сценария для политики Restricted;
- результат проверки сценария для политики Unrestricted;
- результат проверки подписанного сценария для политики Undefined;
- результат проверки подписанного сценария для политики Bypass;
- результат проверки подписанного сценария для политики AllSigned;
- результат проверки подписанного сценария для политики

RemoteSigned.

Должен быть сформирован файл-сценарий, содержащий:

- команду первичной проверки параметра политики выполнения сценариев;

- команду изменения политики выполнения сценариев на каждый из параметров:

- Restricted;
- Unrestricted;
- Undefined;
- Bypass;
- AllSigned;
- RemoteSigned.

Проверка политик выполнения сценариев на удаленном сервере.

Должен быть показан в виде снимка экрана результат проверки подписанного сценария для политики RemoteSigned.

Должен быть сформирован файл-сценарий, содержащий команду изменения политики выполнения сценариев для параметра RemoteSigned.

9 Подготовка сценария для автоматизированного развертывания виртуальных машин

Для автоматизации процесса развертывания ВМ на стенде были разработаны сценарии, которые определяют конфигурацию ВМ после их создания, а также режимы их работы.

Сценарий описывает аппаратную конфигурацию ВМ, которая была определена при создании ВМ в разделе 3. Аппаратная конфигурация подразумевает наличие у ВМ следующих устройств:

- виртуальный жесткий диск;
- видеоадаптер;
- сетевой адаптер.

Также в сценарии задается режим работы ВМ `immutable` – это означает, что ВМ возвращается в исходное состояние при закрытии VirtualBox. Это сделано для того, чтобы ВМ автоматически сбрасывались в исходное состояние по окончании лабораторных работ.

Для автоматизированного развертывания виртуальных машин было подготовлено четыре сценария. Разработанные сценарии представлены в приложении И.

ЗАКЛЮЧЕНИЕ

В результате выполнения ВКР был разработан практикум по применению методов и способов защиты компонентов виртуальной инфраструктуры, состоящий из двух заданий. Первое задание практикума посвящено применению политик выполнения PowerShell для управления запуском сценариев. Второе задание практикума посвящено использованию сертификатов для управления запуском сценариев PowerShell.

В процессе дипломного проектирования был произведен обзор развития оболочки PowerShell, а также существующих платформ и технологий виртуализации, были созданы виртуальные машины, входящие в состав виртуального стенда. Были созданы командные файлы для автоматизированного развертывания ВМ на компьютере, определен общий порядок выполнения заданий практикума, разработано описание выполнения каждого задания практикума, разработаны контрольные экземпляры отчетов о выполнении заданий практикума, определен порядок контроля выполнения заданий практикума и разработаны сценарии для развертывания ВМ.

В результате выполнения заданий практикума студенты получают навыки по применению механизмов обеспечения безопасности, реализованных в оболочке Windows PowerShell с использованием сценариев.

Таким образом, все пункты технического задания на выпускную квалификационную работу выполнены полностью.

Результаты ВКР были представлены на IV Ежегодной межвузовской научно-практической конференции «Информационные технологии в науке и образовании. Проблемы и перспективы» (МНПК – 2017) в секции «Информационная безопасность систем и технологий». Публикации, содержащие представленные результаты, включены в сборник статей [8] конференции, находящийся в издании.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Введение в Windows PowerShell [Электронный ресурс]. Режим доступа: <http://greendail.ru/book/116>, свободный. – Загл. с экрана.
2. ИТ-революция: Хроники 1904-2014: Коммуникации, Интернет, Масс Медиа [Электронный ресурс]. Режим доступа: https://books.google.ru/books/about/ИТ_революция_Хроники_1.html, свободный. – Загл. с экрана.
3. Hyper-V - MSDN – Microsoft [Электронный ресурс]. Режим доступа: [https://msdn.microsoft.com/ru-ru/library/hh831531\(v=ws.11\).aspx](https://msdn.microsoft.com/ru-ru/library/hh831531(v=ws.11).aspx), свободный. – Загл. с экрана.
4. Виртуализация VMware для настольных компьютеров, серверов, приложений, общедоступных и гибридных облаков [Электронный ресурс]. Режим доступа: <https://www.vmware.com/ru.html>, свободный. – Загл. с экрана.
5. Xen – xgu.ru [Электронный ресурс]. Режим доступа: <http://xgu.ru/wiki/Xen>, свободный. – Загл. с экрана.
6. KVM – xgu.ru [Электронный ресурс]. Режим доступа: <http://xgu.ru/wiki/KVM>, свободный. – Загл. с экрана.
7. Загрузить VirtualBox – Oracle [Электронный ресурс]. Режим доступа: <http://www.oracle.com/technetwork/ru/servers-storage/virtualbox/downloads/index.html>, свободный. – Загл. с экрана.
8. Фатеев А.Г., Шилов Б.С. Особенности обеспечения безопасности компонентов виртуализации инфраструктур. Информационные технологии в науке и образовании. Проблемы и перспективы : сб. науч. ст. IV Ежегод. межвуз. науч.-практ. конф. (г. Пенза, 15 марта 2017 года) / под. ред. Л.Р. Фионовой, А.В. Дурина (в печати).

ПРИЛОЖЕНИЕ А

(обязательное)

Структура виртуального стенда

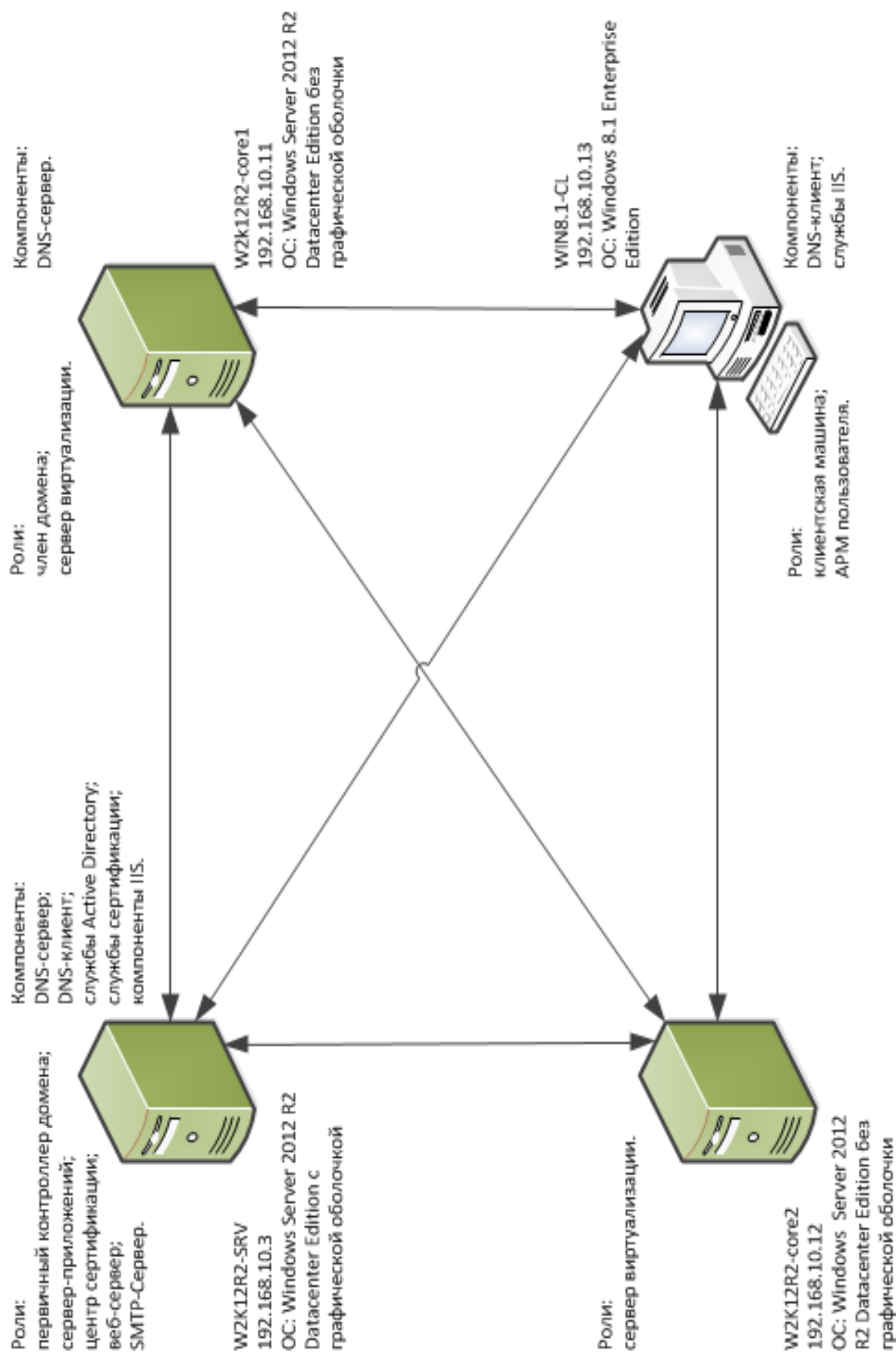


Рисунок А.1

ПРИЛОЖЕНИЕ Б

(обязательное)

UML–диаграмма последовательности действий выполнения
практического задания по применению политик выполнения PowerShell для
управления запуском сценариев

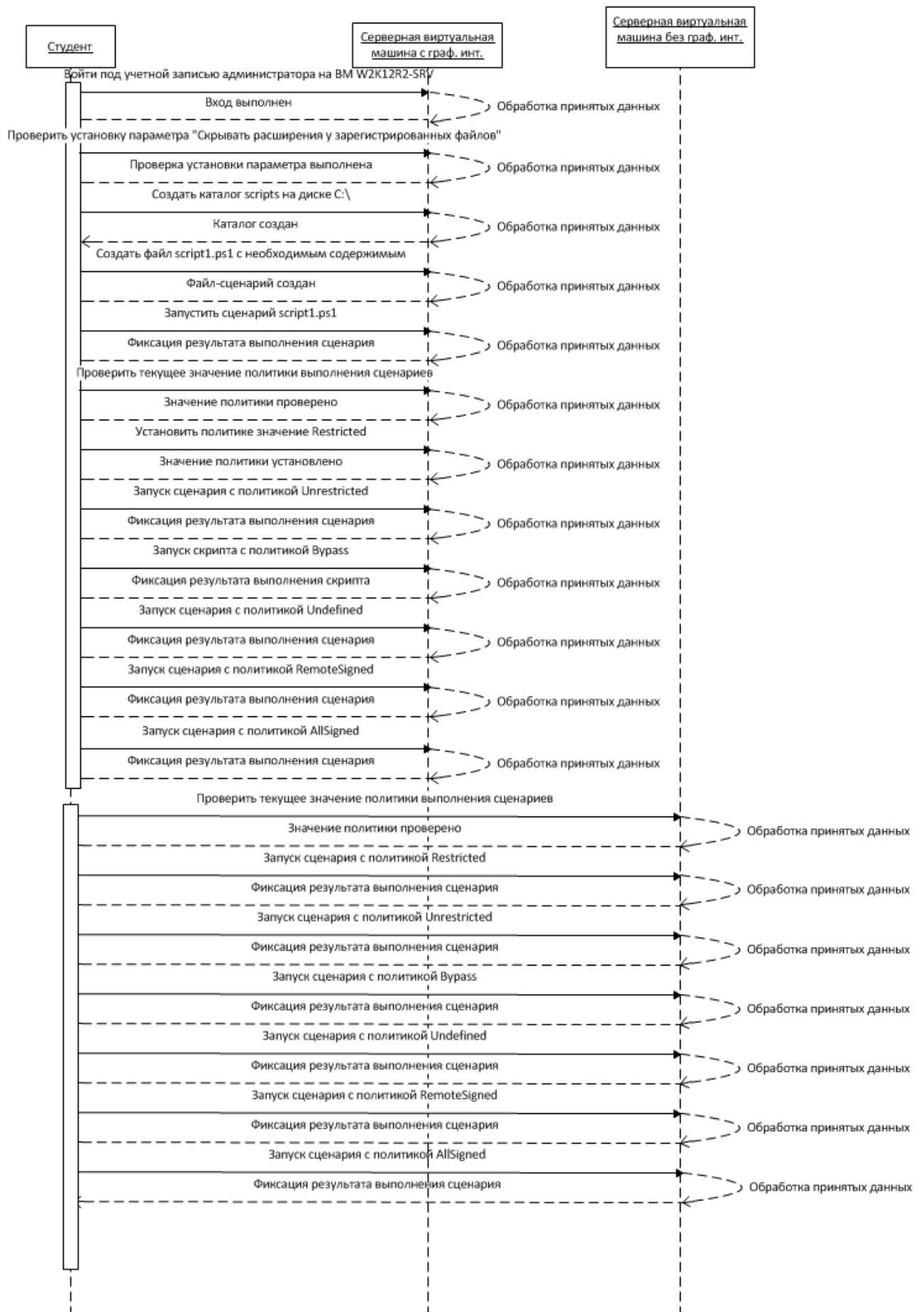


Рисунок Б.1

ПРИЛОЖЕНИЕ В

(обязательное)

UML–диаграмма последовательности действий выполнения
практического задания по использованию сертификатов для управления
запуском сценариев PowerShell

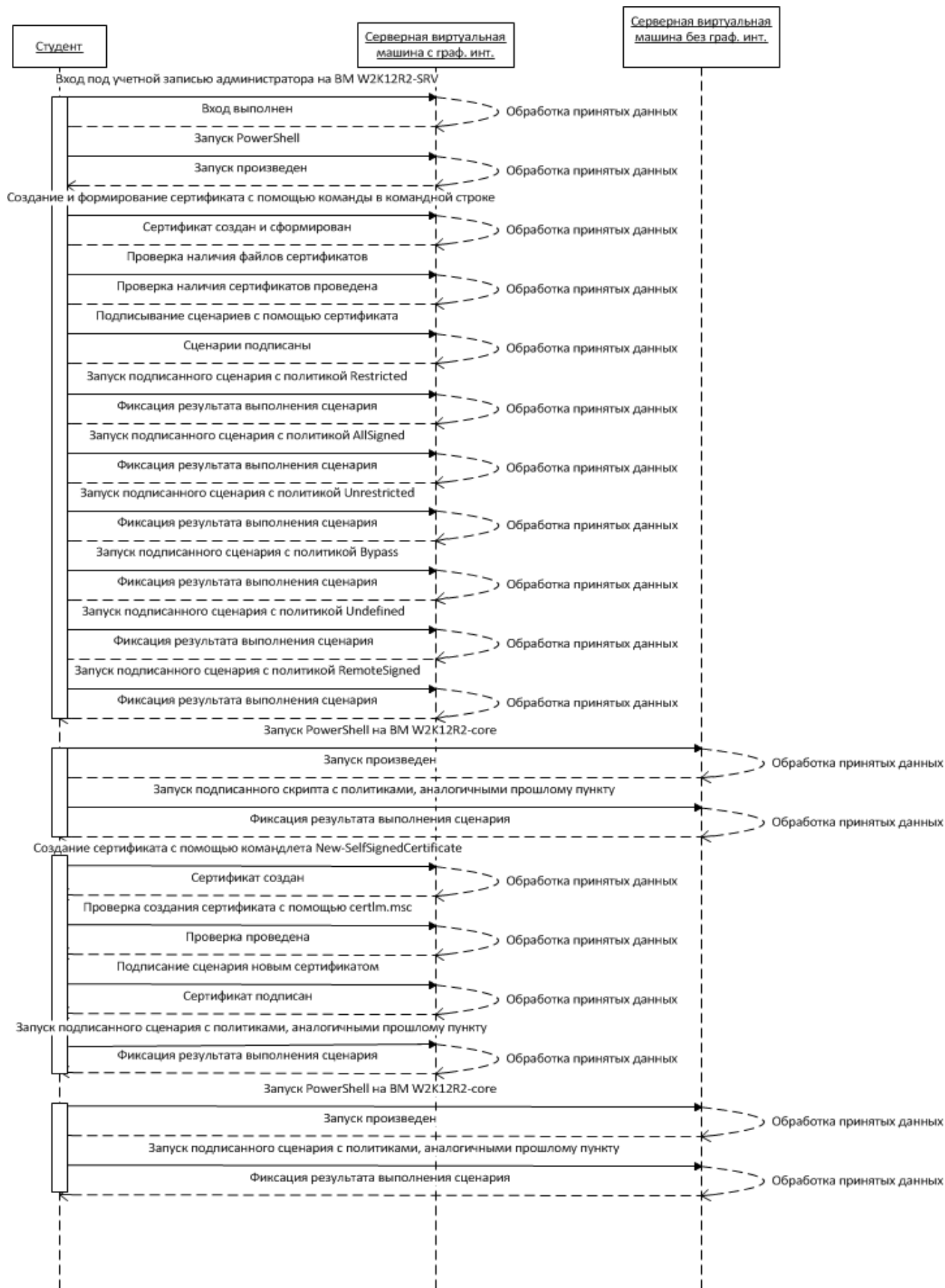


Рисунок В.1

ПРИЛОЖЕНИЕ Г
(обязательное)

Структура отчета о выполнении заданий практикума



Рисунок Г.1

ПРИЛОЖЕНИЕ Д

(обязательное)

Контрольный экземпляр отчета по выполнению задания по изучению применения политик выполнения PowerShell для управления запуском сценариев

Министерство образования и науки РФ
ФГБОУ ВО «ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
Кафедра «Информационная безопасность систем и технологий»

ОТЧЕТ

о выполнении лабораторной работы № (указывается номер лабораторной
работы)

«Название лабораторной работы»

Дисциплина:

Группа:

Выполнил: И.О. Фамилия

Количество баллов:

Дата сдачи:

Принял:

Пенза, год

Цель работы - изучить основные значения политик выполнения сценариев в оболочке Windows PowerShell, а также получить навыки по применению этих политик для управления запуском сценариев

Ход работы

1 Подготовительные действия для начала выполнения задания практикума

1.1 Создание файла-сценария с необходимым кодом

1.2 Созданный файл прилагается к отчету

2 Локальная проверка настроек политики

2.1 Запуск оболочки Windows PowerShell

2.2 Последовательная установка настроек политики и запуск сценария при каждой из настроек

2.3 Фиксация результатов выполнения сценария в виде снимков экрана при каждом значении политики.

3 Проверка настроек политики на удаленном сервере

3.1 Запуск оболочки Windows PowerShell на сервере без графического интерфейса

3.2 Последовательная установка настроек политики и запуск сценария при каждой из настроек

3.3 Фиксация результатов выполнения сценария в виде снимков экрана при каждом значении политики.

Вывод:

ПРИЛОЖЕНИЕ Е

(обязательное)

Контрольный экземпляр отчета по выполнению задания по изучению применения политик выполнения PowerShell для управления запуском сценариев

Министерство образования и науки РФ
ФГБОУ ВО «ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
Кафедра «Информационная безопасность систем и технологий»

ОТЧЕТ

о выполнении лабораторной работы № (указывается номер лабораторной
работы)

«Название лабораторной работы»

Дисциплина:

Группа:

Выполнил: И.О. Фамилия

Количество баллов:

Дата сдачи:

Принял:

Пенза, год

Цель работы - изучить процесс создания, формирования и установки сертификата с помощью makecert, а также с помощью штатных средств оболочки Windows PowerShell. Проверить влияние значений политик выполнения сценариев на подписанные сценарии.

1 Подписывание сценариев с помощью makecert

1.1 Запуск оболочки Windows PowerShell

1.2 Создание сертификата с помощью makecert

1.3 Формирование сертификата с помощью makecert.

1.4 Подписание сертификата с помощью makecert

2 Проверка настроек политики выполнения сценариев на локальном сервере

2.1 Последовательная установка настроек политики и запуск сценария при каждой из настроек

2.2 Фиксация результатов выполнения сценария в виде снимков экрана при каждом значении политики.

3 Проверка настроек политики выполнения сценариев на удаленном сервере

3.1 Запуск оболочки Windows PowerShell на сервере без графического интерфейса

3.2 Последовательная установка настроек политики и запуск сценария при каждой из настроек

3.3 Фиксация результатов выполнения сценария в виде снимков экрана при каждом значении политики.

4 Подписывание сценариев с помощью штатных средств оболочки Windows PowerShell

4.1 Запуск оболочки Windows PowerShell

4.2 Создание сертификата с помощью штатных средств оболочки Windows PowerShell

4.3 Проверка создания сертификата и подписание файла сценария с

помощью него

4.4 Проверка работоспособности политик выполнения сценариев с помощью подписанного сценария

Вывод:

ПРИЛОЖЕНИЕ Ж
(обязательное)

Порядок проверки выполнения заданий практикума

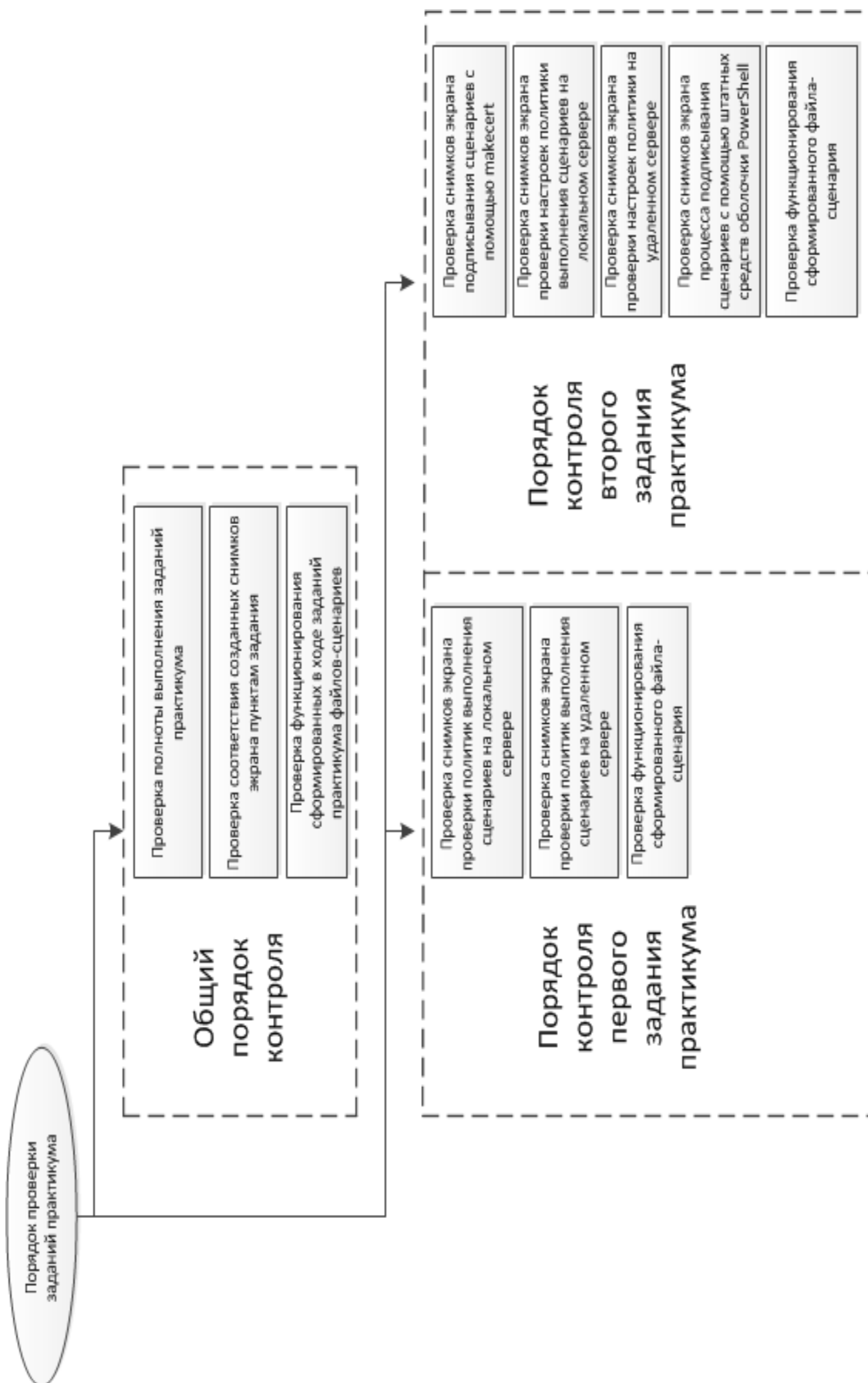


Рисунок Ж.1

ПРИЛОЖЕНИЕ И
(обязательное)

Сценарии для автоматизированного развертывания ВМ

И.1 Сценарий для развертывания ВМ W2K12R2-SRV

@echo off

*rem ******

rem Universal script for VMs

*rem ******

set VHDTYPE="immutable"

set VHDPATH="C:\\W2K12R2-SRV.vdi"

set VMNAME="W2K12R2-SRV"

set OS_TYPE="Windows2012_64"

REM set OS_TYPE="Windows"

set MEMORY="2048"

set VIDEO="64"

set _ACPI="on"

set _IOAPIC="on"

set _PAE="on"

set _HWVIRTEX="on"

set _VTXVID="on"

set netadapter="Intel PRO/1000 VT Server (82545EM)"

set SHAREPATH="D:\\share"

REM if == 0 -> not shared folders

set VBOXM="%programfiles%\\Oracle\\VirtualBox\\vboxmanage"

%VBOXM% createvm --name %VMNAME% --ostype %OS_TYPE% --register

%VBOXM% storagectl %VMNAME% --add ide --controller PIIX4 --name "PIIX4"

*%VBOXM% storageattach %VMNAME% --storagectl PIIX4 --port 0 --device 0 --type hdd -
-medium %VHDPATH% --mtype %VHDTYPE%*

*%VBOXM% storageattach %VMNAME% --storagectl PIIX4 --port 0 --device 1 --type
dvddrive --medium "emptydrive"*

*%VBOXM% modifyvm %VMNAME% --memory %MEMORY% --vram %VIDEO% --acpi
%_ACPI% --ioapic %_IOAPIC% --pae %_PAE% --hwvrtex %_HWVIRTEX% --vtxvpid*

%_VTXVID% --audio none

%VBOXM% modifyvm %VMNAME% --usb on --usbhci on

%VBOXM% modifyvm %VMNAME% --clipboard bidirectional

%VBOXM% modifyvm %VMNAME% --nic1 intnet

*if not %SHAREPATH%=="0" %VBOXM% sharedfolder add %VMNAME% --name share --
hostpath %SHAREPATH%*

@pause

И.2 Сценарий для развертывания ВМ W2K12R2-core1

@echo off

*rem ******

rem Universal script for VMs

*rem ******

set VHDTYPE="immutable"

set VHDPATH="C:\\W2K12R2-core1.vdi"

set VMNAME="W2K12R2-core1"

set OS_TYPE="Windows2012_64"

REM set OS_TYPE="Windows"

set MEMORY="1024"

set VIDEO="64"

set _ACPI="on"

set _IOAPIC="on"

set _PAE="on"

set _HWVIRTEX="on"

set _VTXVID="on"

set netadapter="Intel PRO/1000 VT Server (82545EM)"

set SHAREPATH="D:\\share"

REM if == 0 -> not shared folders

set VBOXM="%programfiles%\\Oracle\\VirtualBox\\vboxmanage"

%VBOXM% createvm --name %VMNAME% --ostype %OS_TYPE% --register

%VBOXM% storagectl %VMNAME% --add ide --controller PIIX4 --name "PIIX4"

*%VBOXM% storageattach %VMNAME% --storagectl PIIX4 --port 0 --device 0 --type hdd -
-medium %VHDPATH% --mtype %VHDTYPE%*

*%VBOXM% storageattach %VMNAME% --storagectl PIIX4 --port 0 --device 1 --type
dvddrive --medium "emptydrive"*

*%VBOXM% modifyvm %VMNAME% --memory %MEMORY% --vram %VIDEO% --acpi
%_ACPI% --ioapic %_IOAPIC% --pae %_PAE% --hwvrtex %_HWVIRTEX% --vtxvpid*

%_VTXVID% --audio none

%VBOXM% modifyvm %VMNAME% --usb on --usbhci on

%VBOXM% modifyvm %VMNAME% --clipboard bidirectional

%VBOXM% modifyvm %VMNAME% --nic1 intnet

*if not %SHAREPATH%=="0" %VBOXM% sharedfolder add %VMNAME% --name share --
hostpath %SHAREPATH%*

@pause

И.3 Сценарий для развертывания ВМ W2K12R2-core2

@echo off

*rem ******

rem Universal script for VMs

*rem ******

set VHDTYPE="immutable"

set VHDPATH="C:\\W2K12R2-core2.vdi"

set VMNAME="W2K12R2-core2"

set OS_TYPE="Windows2012_64"

REM set OS_TYPE="Windows"

set MEMORY="1024"

set VIDEO="64"

set _ACPI="on"

set _IOAPIC="on"

set _PAE="on"

set _HWVIRTEX="on"

set _VTXVID="on"

set netadapter="Intel PRO/1000 VT Server (82545EM)"

set SHAREPATH="D:\\share"

REM if == 0 -> not shared folders

set VBOXM="%programfiles%\\Oracle\\VirtualBox\\vboxmanage"

%VBOXM% createvm --name %VMNAME% --ostype %OS_TYPE% --register

%VBOXM% storagectl %VMNAME% --add ide --controller PIIX4 --name "PIIX4"

*%VBOXM% storageattach %VMNAME% --storagectl PIIX4 --port 0 --device 0 --type hdd -
-medium %VHDPATH% --mtype %VHDTYPE%*

*%VBOXM% storageattach %VMNAME% --storagectl PIIX4 --port 0 --device 1 --type
dvddrive --medium "emptydrive"*

*%VBOXM% modifyvm %VMNAME% --memory %MEMORY% --vram %VIDEO% --acpi
%_ACPI% --ioapic %_IOAPIC% --pae %_PAE% --hwvrtex %_HWVIRTEX% --vtxvpid*

%_VTXVID% --audio none

%VBOXM% modifyvm %VMNAME% --usb on --usbhci on

%VBOXM% modifyvm %VMNAME% --clipboard bidirectional

%VBOXM% modifyvm %VMNAME% --nic1 intnet

*if not %SHAREPATH%=="0" %VBOXM% sharedfolder add %VMNAME% --name share --
hostpath %SHAREPATH%*

@pause

И.4 Сценарий для развертывания ВМ WIN8.1-CL

```
@echo off
rem *****
rem Universal script for VMs
rem *****

set VHDTYPE="immutable"
set VHDPATH="C:\\WIN8.1-CL.vdi"

set VMNAME=" WIN8.1-CL "
set OS_TYPE="Windows8.1_64"
REM set OS_TYPE="Windows"

set MEMORY="2048"
set VIDEO="64"
set _ACPI="on"
set _IOAPIC="on"
set _PAE="on"
set _HWVIRTEX="on"
set _VTXVID="on"
set netadapter="Intel PRO/1000 VT Desktop (82540EM)"

set SHAREPATH="D:\\share"
REM if == 0 -> not shared folders

set VBOXM="%programfiles%\\Oracle\\VirtualBox\\vboxmanage"
%VBOXM% createvm --name %VMNAME% --ostype %OS_TYPE% --register
%VBOXM% storagectl %VMNAME% --add ide --controller PIIX4 --name "PIIX4"

%VBOXM% storageattach %VMNAME% --storagectl PIIX4 --port 0 --device 0 --type hdd -
-medium %VHDPATH% --mtype %VHDTYPE%
%VBOXM% storageattach %VMNAME% --storagectl PIIX4 --port 0 --device 1 --type
dvddrive --medium "emptydrive"
%VBOXM% modifyvm %VMNAME% --memory %MEMORY% --vram %VIDEO% --acpi
%_ACPI% --ioapic %_IOAPIC% --pae %_PAE% --hwvrtex %_HWVIRTEX% --vtxvpid
```

%_VTXVID% --audio none

%VBOXM% modifyvm %VMNAME% --usb on --usbhci on

%VBOXM% modifyvm %VMNAME% --clipboard bidirectional

%VBOXM% modifyvm %VMNAME% --nic1 intnet

*if not %SHAREPATH%=="0" %VBOXM% sharedfolder add %VMNAME% --name share --
hostpath %SHAREPATH%*

@pause