4 Gruppenrichtlinien filtern

In diesem Kapitel werden folgende Themen behandelt:

- Wann bietet sich welche Art von Gruppenrichtlinienfilterung an?
- Sicherheitsfilterungen auf GPOs anwenden
- Wie verwende ich Berechtigungen, um Benutzer vor Gruppenrichtlinienobjekten zu schützen?
- Zwischen verschiedenen Rechnern unterscheiden: mit WMI-Filtern Hardware- und Softwareeigenschaften abfragen

4.1 Einführung

GPOs werden auf Benutzer oder Computer angewendet, weil sich diese im Wirkungsbereich der GPO befinden, also unterhalb der Domäne, der Organisationseinheit oder dem Standort der GPO. Wenn Sie die Wirkungen für GPOs für einzelne Benutzer oder Computer nicht anwenden möchten, müssen Sie die GPO filtern. Dafür stehen Ihnen drei Methoden zur Verfügung.

1. Verweigern Sie einzelnen Benutzern oder Computern das Recht, die Gruppenrichtlinie anzuwenden (negative Sicherheitsfilterung).
2. Erlauben Sie nur einer eingeschränkten Gruppe das Anwenden der Gruppenrichtlinien (positive Sicherheitsfilterung).
3. Ermitteln Sie mithilfe von WMI, ob eine Gruppenrichtlinie angewendet werden soll (WMI-Filter).

Die Filterung über Verweigern ist vor allem dann die Methode der Wahl, wenn es darum geht, ausgewähltes Personal vor der Anwendung eines GPO zu schützen. Die Sicherheitsfil-

terung bietet sich für GPOs an, die nur auf besondere Benutzer oder Computer angewendet werden sollen. WMI-Filter werden in der Regel verwendet, wenn zwischen verschiedenen Rechnern unterschieden werden muss, z. B. zwischen mobilen Systemen und Standgeräten oder zwischen verschiedenen Betriebssystemen.

4.2 Filtern über Gruppenzugehörigkeiten

4.2.1 Berechtigungen verweigern

Wenn Sie verhindern wollen, dass bestimmte Benutzer von den Einstellungen eines GPOs betroffen werden, obwohl das Benutzerkonto im Wirkungsbereich gespeichert ist, können Sie der Gruppe das Recht zum Übernehmen der Einstellungen verweigern. Gehen Sie hierzu folgendermaßen vor:

- Navigieren Sie in der GPMC zu der Gruppenrichtlinie, für die Sie Filterung einrichten möchten, markieren Sie diese und öffnen Sie das Register DELEGIERUNG.

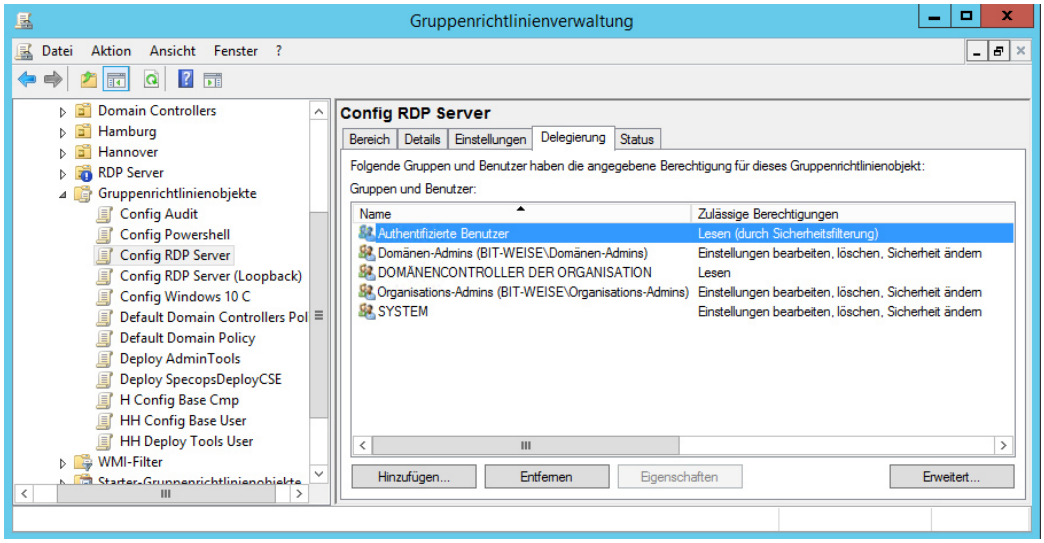


Bild 4.1 Delegierung eines GPOs anzeigen

- Verwenden Sie nun die Schaltfläche ERWEITERT, um die Berechtigungen bearbeiten zu können.
- Klicken Sie auf HINZUFÜGEN, um die Gruppe auszuwählen, der Sie die Berechtigung zum Übernehmen der Gruppenrichtlinie verweigern wollen.
- Fügen Sie die Gruppe wie gewohnt hinzu und setzen Sie anschließend die Berechtigung „Gruppenrichtlinie übernehmen“ auf VERWEIGERT.
- Bestätigen Sie mit OK.

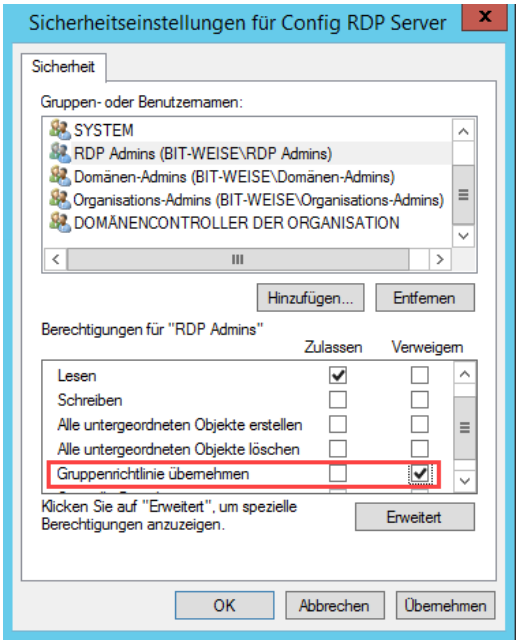


Bild 4.2
Gruppe hinzufügen und Gruppenrichtlinie übernehmen verweigern

Sie können nun im Register DELEGIERUNG des GPO sehen, dass der Gruppe RDP Admins benutzerdefinierte Berechtigungen für das Gruppenrichtlinienobjekt Config RDP Server erteilt wurden. Da es sich nicht um eine Standardberechtigung handelt, sind die effektiven Rechte aber nicht mehr angezeigt. Stattdessen zeigt die Spalte „Zulässige Berechtigungen“ jetzt „Benutzerdefiniert“.

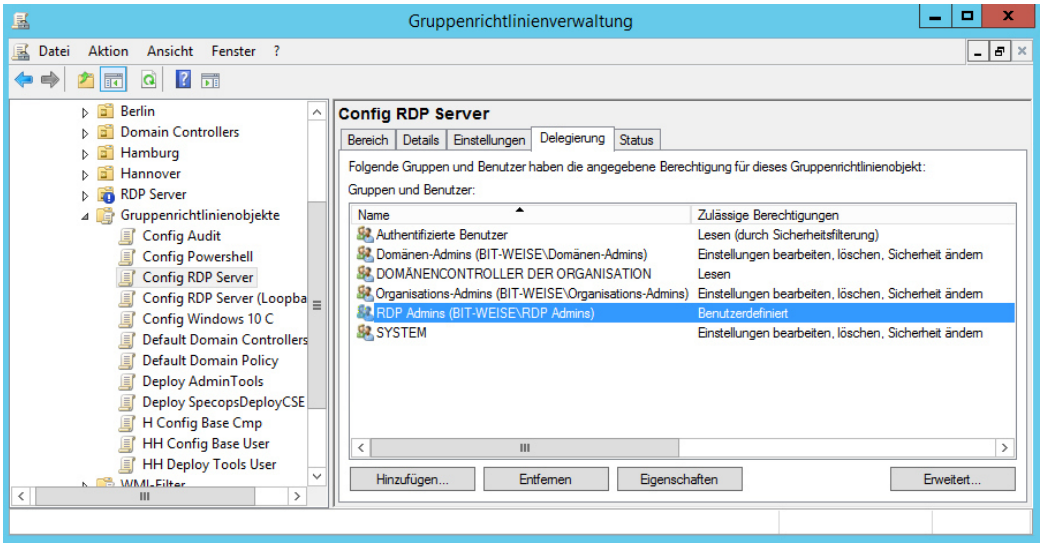


Bild 4.3 Gruppe mit benutzerdefinierter Berechtigung

4.2.2 Sicherheitsfilterung verwenden

Im Register Bereich eines Gruppenrichtlinienobjektes können Sie sehen, für welche Benutzer die Einstellungen gelten. Standardmäßig steht hier die Systemgruppe der authentifizierten Benutzer. Die Gruppe der authentifizierten Benutzer umfasst alle Benutzer, die sich angemeldet haben. Das beinhaltet sogar Benutzer, die sich lokal an Ihrem PC angemeldet haben!

Bevor Sie die nachfolgenden Änderungen durchführen, stellen Sie sicher, dass die Gruppe der Domänen-Computer weiterhin Lese-Zugriff auf die GPO bekommt. Microsoft hat mit dem Patch MS16-072 das Verhalten des Gruppenrichtlinienclients dahin angepasst, dass der Abruf der GPOs vom Server jetzt nicht mehr im Benutzerkontext stattfindet, sondern im Kontext des Computerkontos (siehe Kasten in Kapitel 3). Sie müssen dafür lediglich der Gruppe der Domänencomputer Leserechte auf die GPO geben (siehe Abschnitt 4.2.1). Sie können das AD-Schema auch so anpassen, dass die Berechtigungen beim Anlegen automatisch vergeben werden. Den ganzen Prozess hat Mark Heitbrink für Sie unter *http://www.gruppenrichtlinien.de/artikel/gpo-admin-einrichten-gruppenrichtlinien-delegation/* aufgeschrieben.

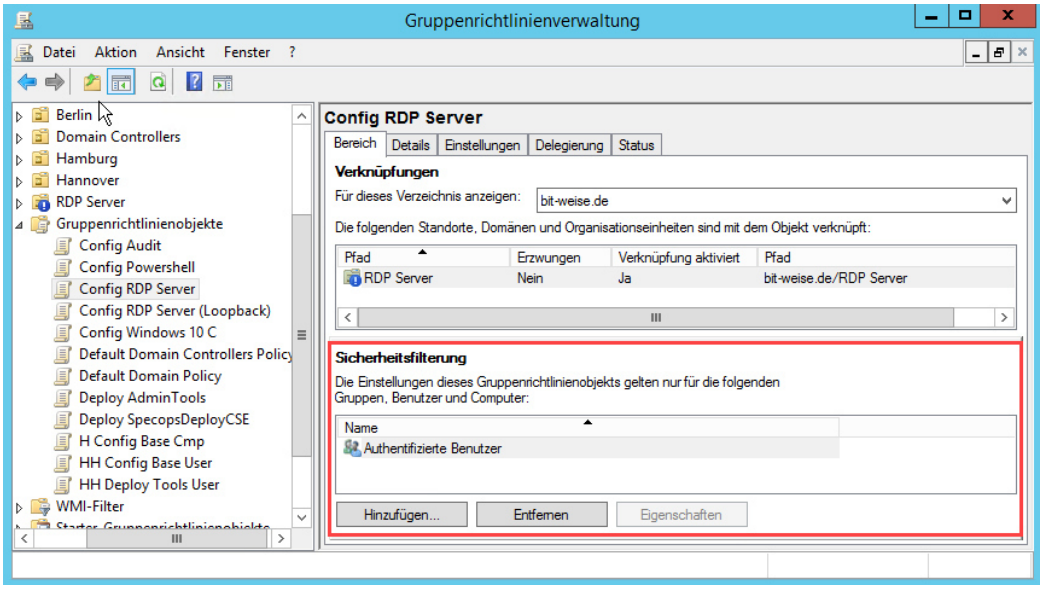


Bild 4.4 Sicherheitsfilterung mit Standardeinstellung

- Klicken Sie Entfernen, um die Gruppe „Authentifizierte Benutzer“ aus der Sicherheitsfilterung zu entfernen.
- Klicken Sie Hinzufügen, um eine Gruppe der Sicherheitsfilterung hinzuzufügen, wählen Sie wie gewohnt die Gruppe aus, und bestätigen Sie mit OK.
- Wenn Sie selber eine Gruppe hinzugefügt haben, können Sie mit der Schaltfläche Eigenschaften die Mitglieder der Gruppe verwalten.

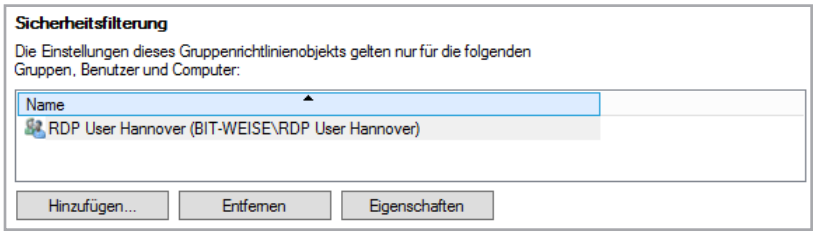


Bild 4.5 Angepasste Sicherheitsfilterung

4.3 WMI-Filter

Mit Windows XP hat Microsoft neben der Sicherheitsfilterung eine zusätzliche Funktion zur Steuerung von Gruppenrichtlinien eingeführt, die WMI-Filter. Der Gruppenrichtlinienclient nutzt WMI-Filter, um den Computer auf bestimmte Eigenschaften wie z. B. die Betriebssystemversion zu prüfen, bevor er eine GPO anwendet. Dadurch wird es möglich, anhand von Computereigenschaften festzulegen, ob eine GPO angewendet wird. Das kann z. B. bei Softwareverteilungsrichtlinien Sinn machen (siehe Kapitel 6), um sicherzustellen, dass der Computer überhaupt das minimal notwendige Betriebssystem besitzt. Oder man kann prüfen, ob ausreichend Festplattenplatz zur Verfügung steht, um eine Anwendung zu installieren. Zusammenfassend kann man sagen, dass WMI-Filter es erlauben, abhängig von den Gegebenheiten des Zielcomputers eine Gruppenrichtlinie anzuwenden.

4.3.1 Einführung in WMI

WMI (Windows Management Instrumentation) ist die Microsoft-Implementierung von WBEM (Web-Based Enterprise Management), einer Initiative mehrerer Hersteller, die Ende der 1990er-Jahre das Ziel hatte, eine einheitliche Verwaltungsplattform ähnlich SNMP (Simple Network Management Protocol) zu schaffen, mit der es möglich sein sollte, Netzwerkgeräte und Computer zentral zu verwalten. Das hat zwar bisher nicht geklappt, u. a. weil Microsoft mit WMI wieder einmal eigene Wege eingeschlagen hat, aber WMI ist zumindest auf jedem Windows-System seit Windows 2000 verfügbar und stellt eine Unmenge von Informationen über den Computer bereit.

WMI teilt die Verwaltungsinformationen in WMI-Klassen ein, die in Form einer Baumstruktur hierarchisch miteinander verbunden sind. Im WMI-Namensraum gibt es jede Menge Zweige, aber für die Verwaltung der Windows-Systeme ist nur ein Zweig vorgesehen: ROOT\CIMV2. CIM steht dabei für Common Infrastructure Model, den allgemeinen Standard, von dem WMI abgeleitet ist (mehr Hintergrund zu WMI finden Sie in der englischen Wikipedia: *https://en.wikipedia.org/wiki/Windows_Management_Instrumentation*).

Im Namensraum (sagen wir der Verständlichkeit halber einfach Ordner) ROOT\CIMV2 finden Sie eine Reihe von Klassen. Diese Klassen haben alle Namen, und alle interessanten

Klassen beginnen mit dem Namen Win32_. Die Win32-WMI-Klassen sind der Ort, an dem die interessanten Informationen über Ihre Computer in Form von Eigenschaften gespeichert sind.[1]

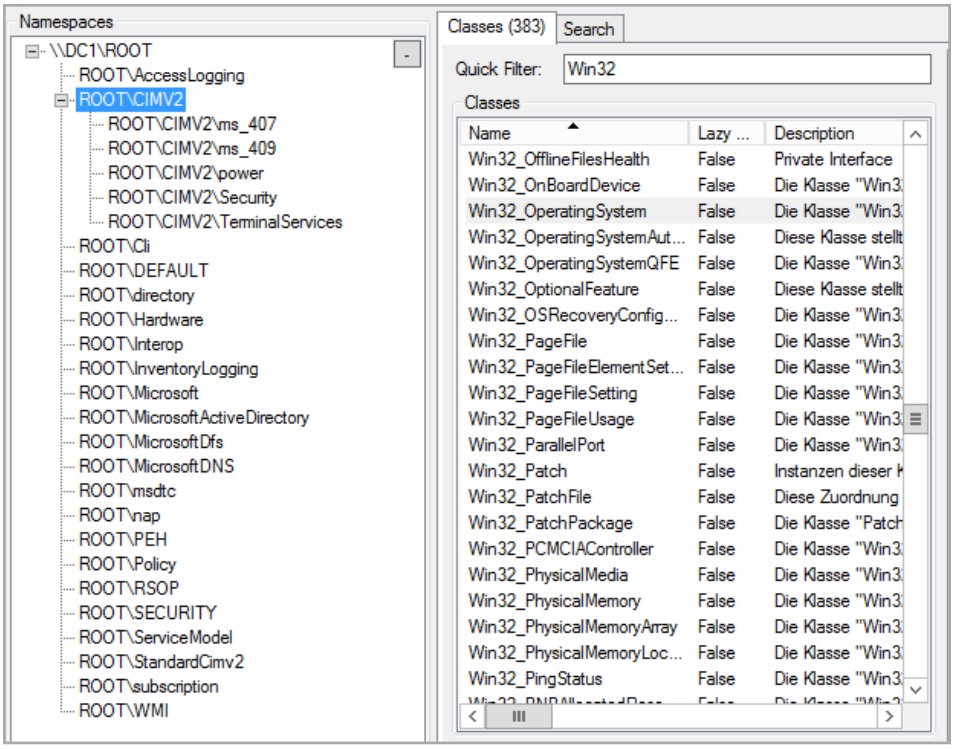


Bild 4.6 Der WMI-Namensraum hat eine Baumstruktur

In den Klassen sind schier unerschöpfliche Informationen gespeichert. Um die Daten aus den Klassen abfragen zu können, stellt Microsoft die WQL (WMI Query Language) zur Verfügung. Sie brauchen jetzt keine Angst davor zu haben, eine neue Programmiersprache lernen zu müssen, denn WQL ist zum einen fast identisch mit SQL, zum anderen brauchen Sie nur eine einfache Select-Abfrage, die Sie einfach variieren können.

Eine WQL-Abfrage hat folgende Form:

Listing 4.1 Die Klasse Win32_OperatingSystem

```
Select * from Win32_OperatingSystem
```

SELECT * FROM besagt, dass Sie alle Eigenschaften der Klasse abfragen möchten, die hinter dem FROM steht.

[1] Ganz korrekt ist das eigentlich nicht, da die Informationen nicht in den Klassen stehen, sondern in den von ihnen abgeleiteten Instanzen, aber die schmutzigen Details sind an dieser Stelle für das weitere Verständnis nicht relevant.

Diese Abfrage können Sie einfach mit PowerShell nachvollziehen, indem Sie das Cmdlet `Get-WMIObject` verwenden (siehe Listing 4.2).

Listing 4.2 Mit PowerShell eine WMI-Abfrage ausführen

```
> Get-WmiObject -Query "Select * from Win32_OperatingSystem"
SystemDirectory : C:\Windows\system32
Organization    :
BuildNumber     : 14393
RegisteredUser  : Windows User
SerialNumber    : 00329-00000-00003-AA690
Version         : 10.0.14393
```

Sie sehen, dass Ihnen die Klasse Win32_OperatingSystem u. a. Informationen darüber gibt, wo Ihr Betriebssystem installiert ist, aber auch dessen Version. Tatsächlich ist das nur ein Bruchteil der Informationen, die in der Klasse Win32_OperatingSystem stehen, denn PowerShell unterdrückt einen großen Teil der Eigenschaften, um Sie nicht mit Daten zu überfluten. Versuchen Sie spaßeshalber einmal, die Ausgabe des obigen Befehls in einen Select-Object * weiterzuleiten (siehe Listing 4.3).

Listing 4.3 Alle Eigenschaften von Win32_OperatingSystem anzeigen

```
> Get-WmiObject -Query "Select * from Win32_OperatingSystem" | select *
```

Ich spare Ihnen an dieser Stelle die komplette Ausgabe, es sind 84 Eigenschaften.

Damit Sie nicht alle Klassen per PowerShell analysieren müssen, verwenden Sie am besten einen grafischen WMI-Browser. Sehr empfehlenswert, weil leistungsfähig und kostenlos, ist z. B. WMI Explorer, den Sie bei Codeplex unter *https://wmie.codeplex.com/* herunterladen können. Der WMI-Explorer ist ein .NET-basiertes Werkzeug und muss nicht installiert werden. Sie brauchen ihn nur zu entpacken und zu starten (siehe Bild 4.7).

Geben Sie zuerst unter (1) den Computer an, mit dem Sie sich verbinden wollen, und klicken Sie auf Connect. Für den lokalen Rechner können Sie einfach einen . (Punkt) eingeben. Der WMI-Explorer fragt jetzt den Namensraum ab und zeigt ihn unter Namespaces an. Wählen Sie als Nächstes unter Namespaces „ROOT\CIMV2“ aus (2) und geben Sie im Feld „Quick Filter“ (3) Win32 ein. Der WMI-Explorer zeigt dann nur noch die Klassen an, die mit Win32 beginnen. Nun haben Sie eine vollständige Auflistung aller Klassen und können drauflos experimentieren. Wählen Sie z. B. Win32_ComputerSystem (4) aus, wird die Auswahl unter Instances aufgelistet und die Eigenschaften der Instanz werden rechts in der Liste angezeigt. Hier finden Sie eine Eigenschaft namens „TotalPhysicalMemory“, die Ihnen den verfügbaren physikalischen Speicher anzeigt. Klicken Sie ruhig mal ein bisschen in den Klassen herum – kaputt machen können Sie nichts, aber es gibt viel Spannendes zu entdecken. Schauen Sie sich z. B. Win32_Bios an. Wenn Sie unter *Classes* die Maus über einer Klasse stehen lassen, wird Ihnen übrigens auch die Beschreibung der Klasse angezeigt (die Sie auch unter *Description* finden).

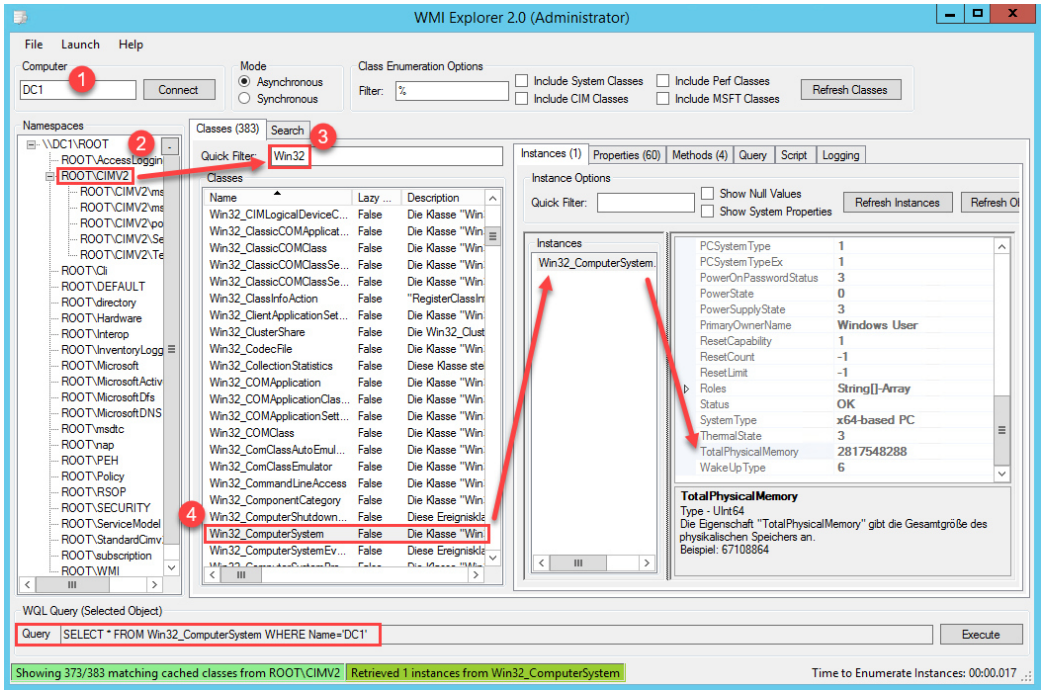


Bild 4.7 WMI-Explorer stellt den WMI-Namensraum grafisch dar.

WQL erlaubt es auch, WMI-Daten auf bestimmte Bedingungen zu überprüfen. Für die Überprüfung implementiert WQL wieder die SQL-Syntax – wer ein bisschen SQL kann, ist also fein raus. Um bestimmte Datensätze auszufiltern, verwendet SQL die Where-Klausel:

```
Select * from Win32_OperatingSystem
Where OSArchitecture = '64-Bit'
```

Diese Abfrage bedeutet übersetzt: Gib mir alle Eigenschaften von Win32_OperatingSystem zurück, wenn (oder wo) die Eigenschaft OSArchitecture dem Wert „64-Bit“ entspricht. Weil der Eintrag „64-Bit“ ein Text ist, muss er zusätzlich in Anführungszeichen gesetzt werden.

Ist der Computer, auf dem Sie die Abfrage ausführen, mit einem 64-Bit-Windows installiert worden, gibt die Abfrage alle Eigenschaften der Klasse Win32_OperatingSystem zurück. Hierfür ist das * hinter dem Select verantwortlich. Ist der Computer ein 32-Bit-System, liefert die Abfrage gar nichts zurück. Mit WQL können Sie aber nicht nur auf Gleichheit prüfen.

Tabelle 4.1 Die Operatoren mit Beispielen

Operator	Bedeutung	Beispiel
>	Größer	`SELECT * FROM Win32_LogicalDisk WHERE (FreeSpace > 5368709120) and (DeviceID = "C:")`
>=	Größer oder gleich	`Select * from Win32_ComputerSystem Where TotalPhysicalMemory >= 2146451456`
<	Kleiner	
<=	Kleiner oder gleich	
<>	Ungleich	`select * from Win32_OperatingSystem WHERE (ProductType <> "2") AND (ProductType <> "3")`
like	Vergleich mit einem Muster. Als Platzhalter wird % verwendet.	`SELECT * FROM Win32_OperatingSystem where (Version like '10.%') or (Version like '6.3.%')`

Wenn Sie eine WQL-Abfrage erstellt haben, können Sie sie im WMI Explorer auch gleich prüfen, indem Sie eine Klasse auswählen und dann das Register „Query“ öffnen. Geben Sie hier Ihre Abfrage im Textfenster „WQL Query“ ein und wählen Sie die Schaltfläche EXECUTE. Ob die Abfrage ein Ergebnis liefert, sehen Sie im Fenster „Results“.

4.3.2 WQL zum Filtern von GPOs

Mit dem Wissen um WMI ist es jetzt einfach, einen WMI-Filter zu schreiben. Ein WMI-Filter ist nämlich nichts anderes als eine im AD hinterlegte WQL-Abfrage, die mit einer GPO verbunden wird. Der WMI-Filter entspricht dabei genau einer WQL-Abfrage. Der Gruppenrichtlinienclient wertet, wenn eine GPO mit einem WMI-Filter verbunden ist, die WQL-Abfrage lokal auf dem Client aus. Wenn die WQL-Abfrage irgendeine Rückgabe liefert, wird die GPO auf dem Client angewendet. Gibt die WQL keinen Wert zurück, wird die GPO übersprungen. Schauen Sie sich dazu noch einmal die WQL-Abfrage an, die den verfügbaren Arbeitsspeicher testet: `Select * from Win32_ComputerSystem Where TotalPhysicalMemory >= 2146451456`. Wenn Sie diese Abfrage auf Ihrem Computer ausführen, gibt die Abfrage alle Eigenschaften der Klasse Win32_ComputerSystem zurück (aufgrund des *), wenn Ihr Computer über mindestens 2 GB RAM verfügt. Hat Ihr Computer weniger als 2 GB RAM, liefert die Abfrage kein Ergebnis, denn es konnte ja keine Klasse gefunden werden, auf die die Where-Bedingung zutrifft. Liefert die Abfrage kein Ergebnis zurück, wird die GPO übersprungen.

4.3.3 WMI-Filter erstellen

- Erweitern Sie in der GPMC die Konsolenstruktur und klicken Sie auf WMI-Filter.
- Wählen Sie im Kontextmenü den Befehl Neu.

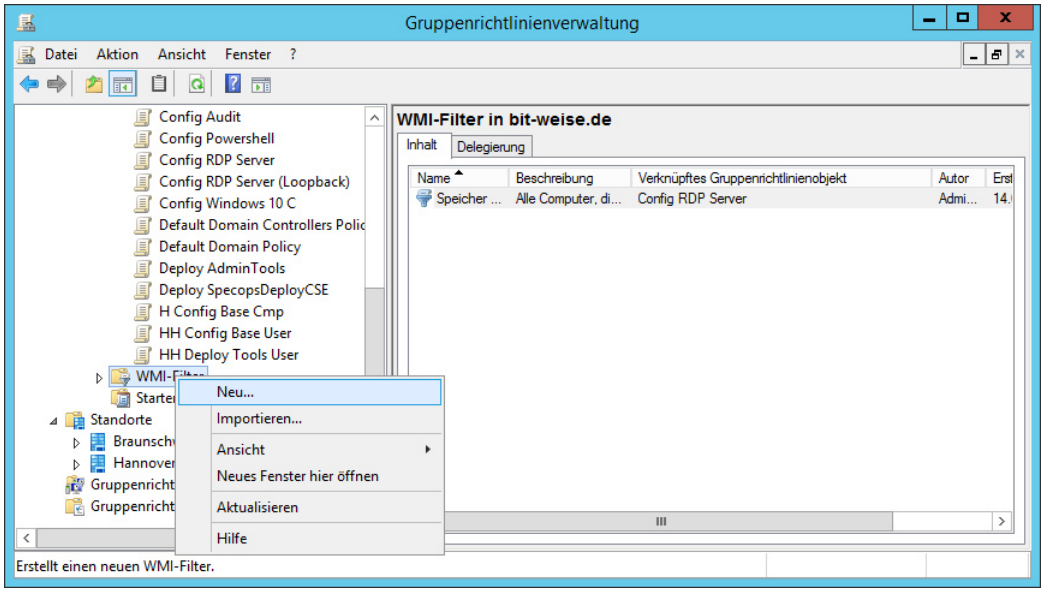


Bild 4.8 Neuen WMI-Filter erstellen

- Geben Sie einen Namen und eine Beschreibung (optional) für den neuen Filter an, z. B. „WMI-Filter für mobile Rechner“.
- Klicken Sie unter Abfragen auf Hinzufügen.
- Belassen Sie den Namespace bei root\CIMv2 und geben Sie bei Abfrage die gewünschte Abfrage ein.

HINWEIS: Im folgenden Beispiel wird eine Abfrage nach dem PCSystemType aus der Klasse Win32_ComputerSystem verwendet, wobei „2“ dem Type Mobile entspricht. PCSystemType steht allerdings erst ab Vista zur Verfügung. Wenn Sie tatsächlich immer noch ältere Systeme im Einsatz haben sollten, verwenden Sie die Klasse Win32_SystemEnclosure und die Eigenschaft ChassisType. Eine schöne Auflistung der möglichen Abfragen finden Sie unter *HTTP://woshub.com/sccm-and-wmi-query-to-find-all-laptops-and-desktops/*.

Listing 4.4 WQL zum Filtern von mobilen Geräten

```
SELECT * FROM Win32_ComputerSystem WHERE PCSystemType = 2
```

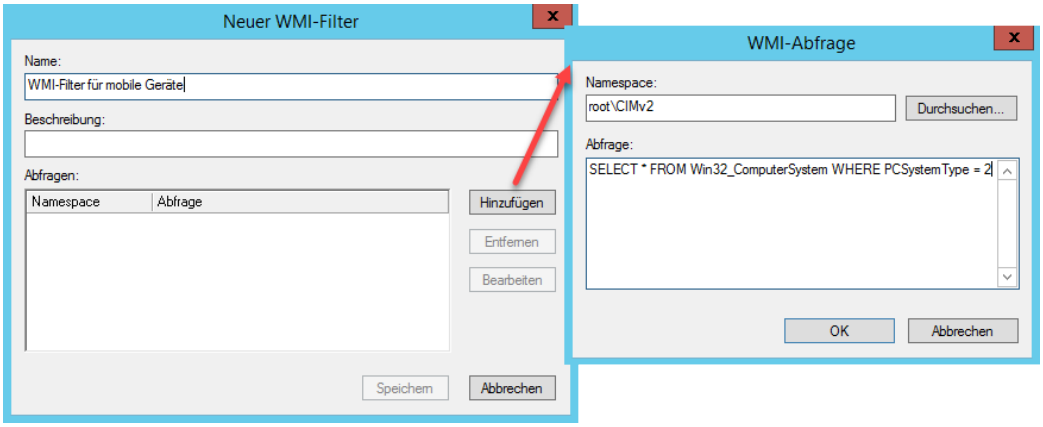
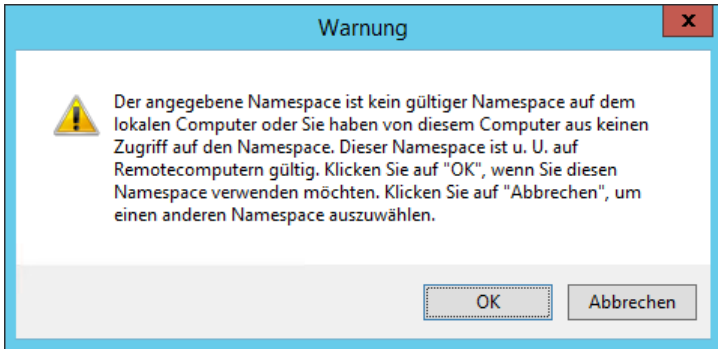


Bild 4.9 WMI-Abfrage definieren

- Bestätigen Sie Ihre Abfragedefinition mit OK.

  Wenn Sie an dieser Stelle eine Warnung erhalten, die besagt, dass „der angegebene Namespace kein gültiger Namespace auf dem lokalen Computer ist“, verwenden Sie vermutlich Windows Server 2012 R2. Brechen Sie sicherheitshalber noch einmal ab, und klicken Sie in Ihrer WMI-Abfrage auf Durchsuchen. Sie bekommen dann alle WMI-Namensräume angezeigt. Sollte Ihnen in der Liste nicht „root\CIMv2“ angezeigt werden, haben Sie vermutlich tatsächlich ein Problem. Wird der Namensraum aufgelistet, können Sie die Warnung einfach ignorieren. Es handelt sich um einen Bug in Windows Server 2012 R2, der WMI-Filter funktioniert trotzdem einwandfrei. Ab Windows Server 2016 erscheint die Warnung nicht mehr.



- Sie können auch mehrere WQL-Abfragen in einem Filter zusammenfassen. Klicken Sie hierfür auf Hinzufügen und wiederholen die Prozedur. Alle angegebenen Filter werden nacheinander ausgeführt. Die Abfragen sind AND-verknüpft, was bedeutet, dass alle Abfragen ein Ergebnis liefern müssen, damit die GPO angewendet wird. Es gibt keine Möglichkeit, dieses Verhalten zu ändern!
- Wenn Sie Speichern auswählen, überprüft das System die WMI-Abfrage. Sollte diese nicht korrekt sein, erhalten Sie eine Fehlermeldung.

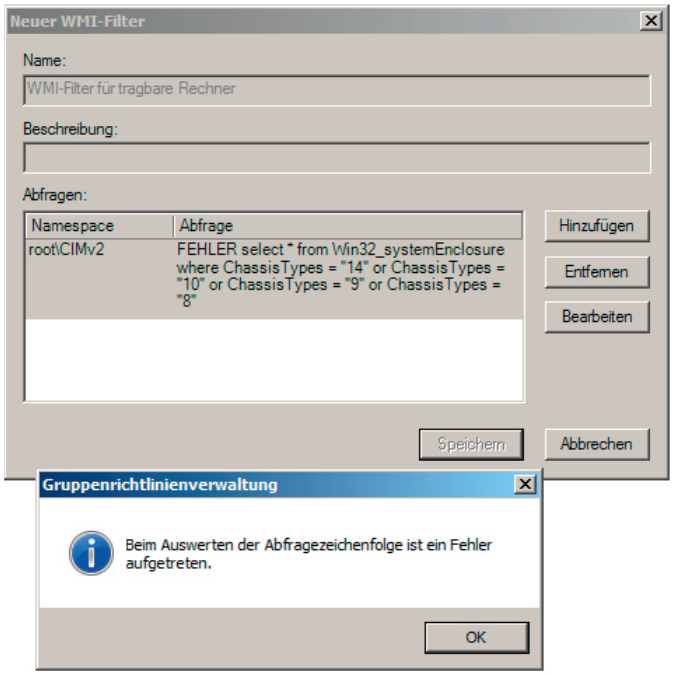


Bild 4.10 Syntaxprüfung

4.3.4 WMI-Filter anwenden

- Markieren Sie die GPO, dem Sie einen WMI-Filter zuweisen möchten, und klicken Sie im unteren Fenster-Bereich unter WMI-Filterung auf das ROLLFELD. Wählen Sie dort den Filter, den Sie verwenden möchten.
- Sie werden gefragt, ob Sie den Filter ändern wollen. Bestätigen Sie mit JA.

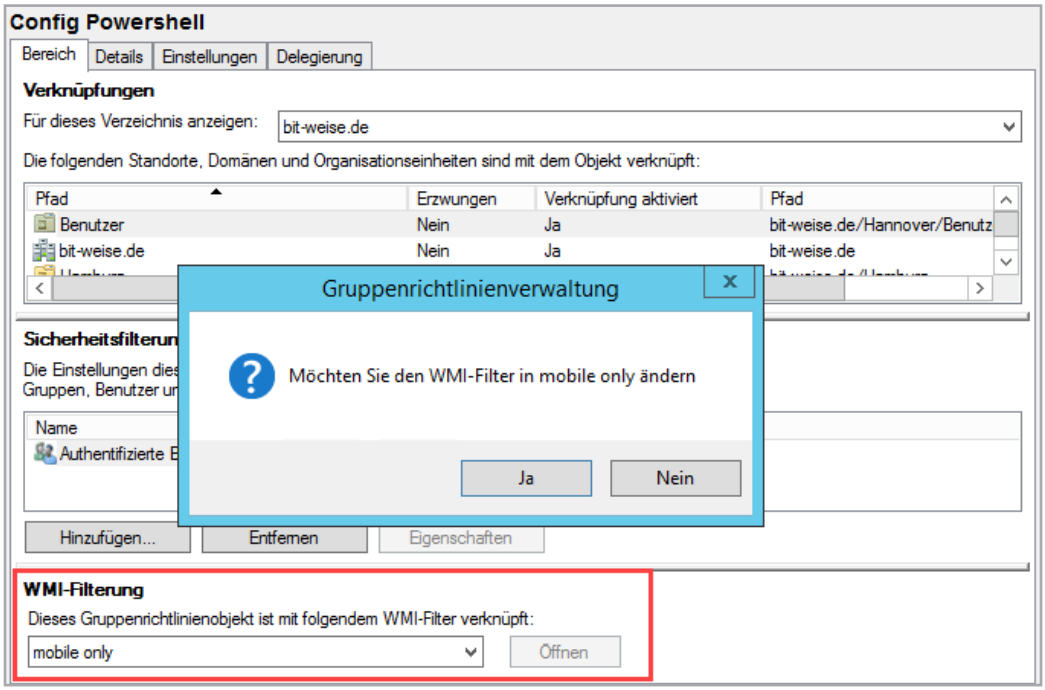


Bild 4.11 WMI-Filter zuweisen

4.3.5 WMI-Filter entfernen

- Markieren Sie die GPO, der Sie einen WMI-Filter zuweisen möchten, und klicken Sie im unteren Fenster-Bereich unter WMI-Filterung auf das Rollfeld. Wählen Sie dort <Kein>, um den Filter zu entfernen.
- Sie werden gefragt, ob Sie den WMI-Filter entfernen möchten. Bestätigen Sie mit Ja.

4.3.6 Beispiele von WMI-Abfragen für WMI-Filter

WMI-Filter lassen sich für eine schier unerschöpfliche Anzahl von Abfragen verwenden. Sie können z. B. Hardwarekomponenten auslesen, installierte Anwendungen oder Treiber abfragen oder auch den Betriebsstatus von Computern auslesen. Eine englische Dokumentation sämtlicher WMI-Klassen finden Sie im Internet unter *http://msdn.microsoft.com/en-us/library/aa394572%28v=VS.85%29.aspx*.

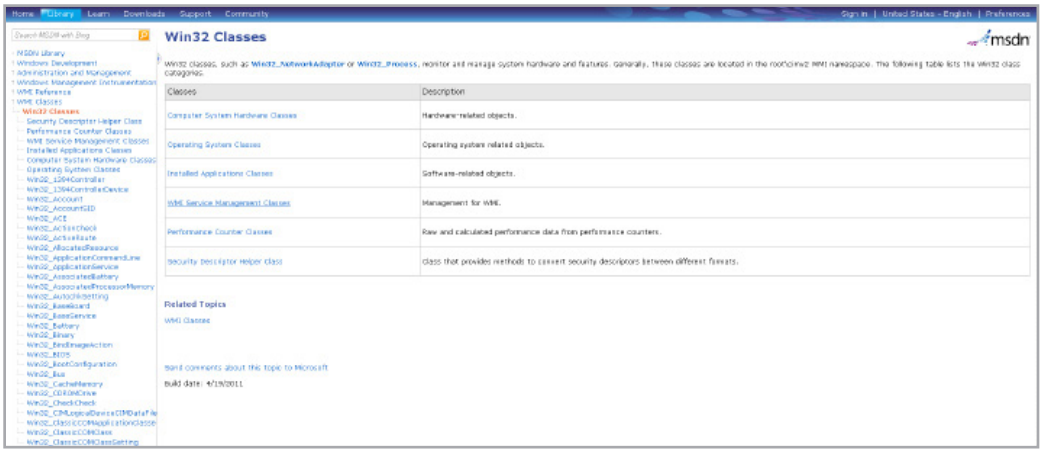


Bild 4.12 Auszug aus den Win32-Klassen

Tabelle 4.2 zeigt Ihnen einige WMI-Abfragen, die Sie verwenden können, um Abfragen in Ihrem Netzwerk zu definieren.

Tabelle 4.2 WMI-Abfragen für WMI-Filter

Abfrage, ob das SP1 für Windows 7 Enterprise installiert ist: `Select Caption from Win32_OperatingSystem where ServicePackMajorVersion <> 0 and Caption like "Microsoft Windows 7 Enterprise%"` In diesem Fall wird nach der Enterprise-Version gesucht, daher reicht es nicht, nach der Betriebssystemversion zu filtern. In der Abfrage wird ein `like` verwendet, da eventuell auf das Enterprise folgender Text bei einem Vergleich mit = kein Ergebnis liefern würde. Die Eigenschaft ServicePackMajorVersion ist 0, wenn kein Servicepack installiert ist. Da es für Windows 7 nur SP1 gibt, wird in der Abfrage auf ungleich 0 geprüft.
Abfrage, ob das installierte Betriebssystem ein deutschsprachiges Windows 7 ist: `Select Version from Win32_OperatingSystem where Version like "6.1%" and CountryCode="49"`
Abfrage, ob das installierte Betriebssystem Windows 8 oder Windows 10 ist: `Select Version from Win32_OperatingSystem where (Version like "6.3%") or (Version like "10.%")`
Abfrage, ob das Rechnermodell ein Laptop FSC Lifebook E8010 mit Intel-Prozessor ist: `Select Model from Win32_ComputerSystem where (manufacturer = "FIJITSU SIEMENS") and (Model = "LIFEBOOK E8010 INT")`
Abfrage, ob Hotfix KB2478063 (Microsoft .NET Framework 4 Platform-Update 1 – Laufzeitupdate) installiert ist: `Select HotFixID from Win32_QuickFixEngineering where HotFixID = "KB2478063"` **ACHTUNG! Diese Abfrage läuft sehr lange und kann die Verarbeitung Ihrer Gruppenrichtlinien deutlich verzögern. Versuchen Sie, das Suchen nach Hotfixes zu vermeiden!**
Abfrage, ob der Firewalldienst läuft: `Select State from Win32_service where name='MpsSvc' and State='Running'`

Abfrage, ob ein Rechner tragbar ist:

```
SELECT PCSystemType FROM Win32_ComputerSystem WHERE PCSystemType = 2
```

Abfrage, ob auf dem Datenträger C: mindestens 5 GB Speicherplatz frei ist:

```
SELECT FreeSpace FROM Win32_LogicalDisk WHERE (FreeSpace > 5368709120)
and (DeviceID = "C:")
```

Abfrage, ob mindestens 2 GB Arbeitsspeicher installiert sind:

```
Select TotalPhysicalMemory from Win32_ComputerSystem Where TotalPhysical
Memory >= 2146451456
```

4.3.7 WMI-Filter optimieren

WMI-Filter sind eine tolle Sache, weil Sie GPOs so anhand der Möglichkeiten der Clients anwenden können. Aber für WMI-Filter gilt wie für alles andere auch: Testen Sie Ihre WMI-Filter, bevor Sie sie anwenden. Ein WMI-Filter funktioniert nämlich nicht unbedingt so, wie Sie es erwarten, und unter Umständen braucht das Abfragen der WMI-Datenbank außerdem auch noch sehr lange. Die meisten WMI-Filter sind zwar in Millisekunden abgearbeitet. Sie können die Verarbeitung sogar noch optimieren, indem Sie in der WQL-Abfrage hinter dem Select nicht * angeben, sondern eine der Eigenschaften der WMI-Klasse. Der WMI-Filter kann dann noch schneller verarbeitet werden. Bei einer einzelnen Abfrage macht das wenig Performancegewinn, muss ein Gruppenrichtlinienclient aber viele WMI-Abfragen verarbeiten, summieren sich auch Millisekunden zu merklichen Zeitspannen.

Es gibt aber einige Klassen, deren Abfragen zu merklichen Verzögerungen führen und die Sie auf jeden Fall meiden sollten. Das ist die im Beispiel oben verwendete Klasse Win32_QuickFixEngineering, die Ihnen installierte Updates anzeigt, und die Klasse Win32_Product, die Ihnen die auf dem Computer installierten Programme zurückliefert. Beide Klassen rufen die Informationen aber selber erst ab, wenn man die WQL-Abfrage startet, und benötigen mehrere Sekunden (!), um die Abfrage zu beenden. Wenn ein Gruppenrichtlinienclient mehrere solcher aufwendigen WQL-Filter auswerten muss, kann sich das schnell zu lähmenden Wartezeiten für den Benutzer addieren.

Um zu testen, wie lange eine WQL-Abfrage benötigt, können Sie das PowerShell-Cmdlet `Measure-Command` einsetzen, das die Laufzeit eines Kommandos bestimmen kann. Rufen Sie dafür die WQL-Abfrage per `Get-WMIObject` auf, und übergeben Sie das Kommando an Measure-Command:

```
> measure-command { Get-WMIObject -Query "Select * from Win32_QuickFixEngineering
where HotFixID = 'KB316 4035'" }

Days              : 0
Hours             : 0
Minutes           : 0
Seconds           : 1
Milliseconds      : 162
Ticks             : 11627903
TotalDays         : 1,34582210648148E-05
TotalHours        : 0,000322997305555556
```

```
TotalMinutes      : 0,0193798383333333
TotalSeconds      : 1,1627903
TotalMilliseconds : 1162,7903
```

Der Abruf der Klasse Win32_QuickFixEngineering hat 1 Sekunde und 162 Millisekunden gebraucht.

Eine gute Untersuchung des Einflusses von WMI-Filtern auf die Anmeldung finden Sie bei Helge Klein unter *https://helgeklein.com/blog/2016/01/how-group-policy-impacts-logon-performance-3-wmi-filters-ilt/*.