Ссылка на статью:
// Ученые записки УлГУ. Серия Математика и информационные технологии. 2024, № 2, с. 91-98.

Поступила: 17.12.2024
Окончательный вариант: 17.12.2024

УДК 0004.932

Подход к инвентаризации системных параметров компьютеров в Windows домене

***Халиков И.И.*[1,*], *Макарова А.Ю.*[1],**
***Подобрий А.Н.*[2,3], *Перцев А.А.*[2,3]**

[*] halikov_ii@mail.ru
[1]УлГУ, Ульяновск, Россия
[2] ФНПЦ АО «НПО «Марс», Ульяновск, Россия
[3]УлГТУ, Ульяновск, Россия

В статье представлен подход к автоматизированной инвентаризации системных параметров компьютеров в домене Windows. Разработанное решение использует PowerShell скрипт, запускаемый по групповым политикам, для сбора информации о характеристиках компьютеров, таких как модель процессора, объем памяти, установленная операционная система, и другие параметры. Собранные данные сохраняются в базе данных MSSQL. Внедрение механизма хэш-записи позволяет отслеживать изменения характеристик компьютеров и оптимизировать частоту записи данных в базу. Предлагаемый подход обеспечивает эффективное и автоматизированное управление ИТ-инфраструктурой, значительно снижая трудозатраты и повышая оперативность принятия решений. Статья описывает архитектуру системы, алгоритмы работы скрипта и структуру базы данных, предоставляя практическое руководство по реализации автоматизированной инвентаризации.

***Ключевые слова**: инвентаризация системных параметров, Windows домен, PowerShell, MSSQL*

Введение

В современном мире IT-инфраструктуры играют ключевую роль в успешном функционировании организаций. Эффективное управление компьютерными ресурсами становится критически важным для обеспечения стабильной работы и безопасности сети. Одним из ключевых аспектов такого управления является инвентаризация системных параметров компьютеров в домене.

Инвентаризация системных параметров компьютеров позволяет получить детальную информацию о каждом устройстве, подключенном к сети, включая его модель, операционную систему, установленные приложения и обновления, а также другие важные параметры. Эта информация не только помогает в управлении активами, но и обеспечивает

основу для принятия обоснованных решений по обновлению и модернизации оборудования, а также для обеспечения безопасности сети.

В открытых источниках [1] представлены программные решения предоставляющие комплексный подход к управлению всеми IT процессами. Функционал данных решений направлен на: обслуживание пользователей (Service Desk), управление ИТ-подразделением (персоналом, имуществом, инфраструктурой, финансами). Однако, широкий функционал сопровождает большую цену для покупателей ПО.

В данной статье, описывается подход, позволяющий автоматизировано производить инвентаризацию характеристик компьютеров в Windows домене.

1. Концептуальная модель

Система инвентаризации характеристик компьютера в домене состоит из 2 основных сегментов, задачи которых сбор и хранение информации.

Первым сегментом является PowerShell скрипт, автоматизировано запускающийся по групповым политикам во время входа доменного пользователя в систему. Скрипт получает информацию о системных параметрах компьютера, проводит обработку и записывает характеристики в MSSQL базу данных. Опрашиваемыми параметрами компьютера являются: серийный номер, доменное имя компьютера, модель процессора, объем оперативной памяти, количество плат оперативной памяти, физические жесткие диски, ip-адрес, mac-адрес, операционная система. Доменное имя является основным параметром, по которому при обращении к базе данных, идентифицируется компьютер, так как его смена проводится крайне редко. Серийный номер не может являться основным идентификатором, так как не у всех компьютеров имеется bios-серийный номер.

Поскольку скрипт запускается при каждом входе пользователя в систему, нет смысла каждый раз записывать системные параметры в базу данных. Поэтому системные параметры конкретного компьютера записываются базу данных только при первом опросе и при изменениях характеристик. Для отслеживания изменений характеристик используется хэш-запись компьютера, который формируется из всех системных параметров при каждом запуске скрипта и записывается в отдельную таблицу базы данных. Хэш-запись нужна для того, чтобы не сравнивать все параметры, а сравнивать только данный параметр, так при любом изменении характеристик компьютера хэш-запись будет уже другой. На рис. 1 представлен алгоритм работы скрипта.

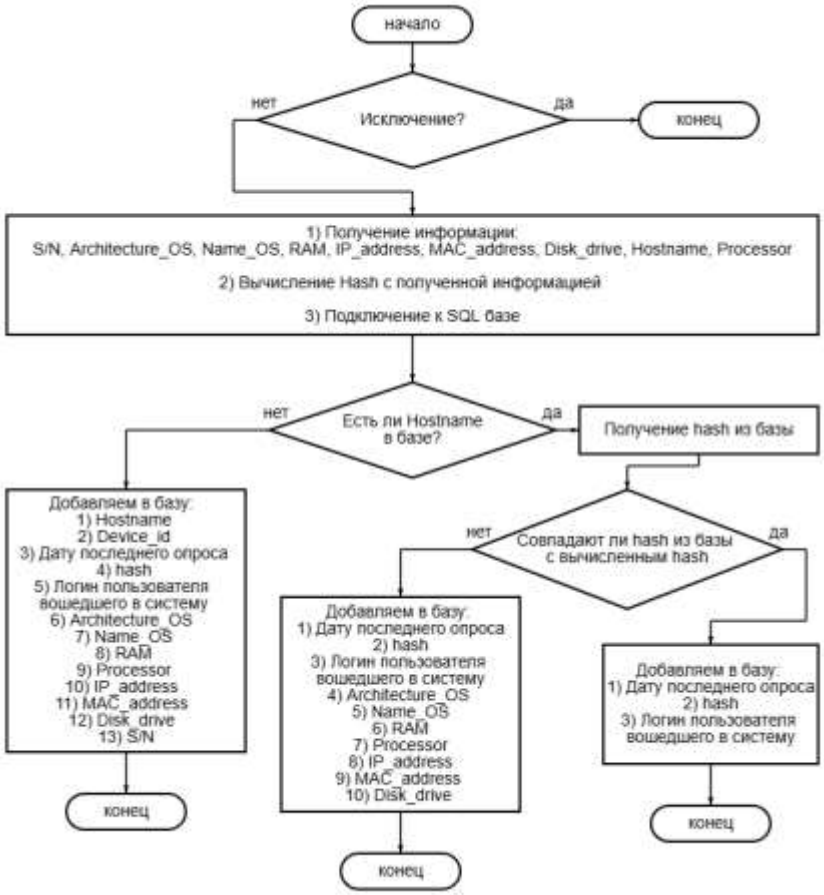

Рис. 1. Алгоритм PowerShell скрипта

MSSQL база данных является вторым сегментом системы, задача которого хранение полученной информации. Исходя из опрашиваемых параметров, сформирована структура БД, состоящая из следующих таблиц:

device — хранится доменное имя и серийный номер компьютера. Формируется идентификатор для каждого компьютера - «device_id»;

device_date_survey — хранится время каждого опроса компьютера (запуска скрипта) для отслеживания актуальности данных. Формируется идентификатор каждого опроса «device_survey_id», сопоставляется с «device_id»;

device_hash — хранится хэш-запись сформированная при каждом запуске скрипта. Формируется идентификатор каждой хэш-записи, сопоставляется с «device_survey_id» и «device_id»;

device_disk — хранится информация о модели, объеме и серийном номере жестких дисков компьютера. Формируется идентификатор каждого жесткого диска «device_disk_id», сопоставляется с «device_survey_id» и «device_id»;

device_processor — хранится информация о модели и тактовой частоте процессора компьютера. Формируется идентификатор каждого процессора «device_processor_id», сопоставляется с «device_survey_id» и «device_id»;

device_network_ip — хранится основной ip-адрес компьютера. Формируется идентификатор каждого ip-адреса «device_network_ip_id», сопоставляется с «device_survey_id» и «device_id»;

device_network_mac — хранится основной mac-адрес компьютера. Формируется идентификатор каждого mac-адреса «device_network_mac_id», сопоставляется с «device_survey_id» и «device_id»;

device_os_system — хранится информация об названии и архитектуре операционной системы компьютера. Формируется идентификатор названия операционной системы «device_os_system_id», сопоставляется с «device_survey_id» и «device_id».

Структура БД представлена на рис. 2.

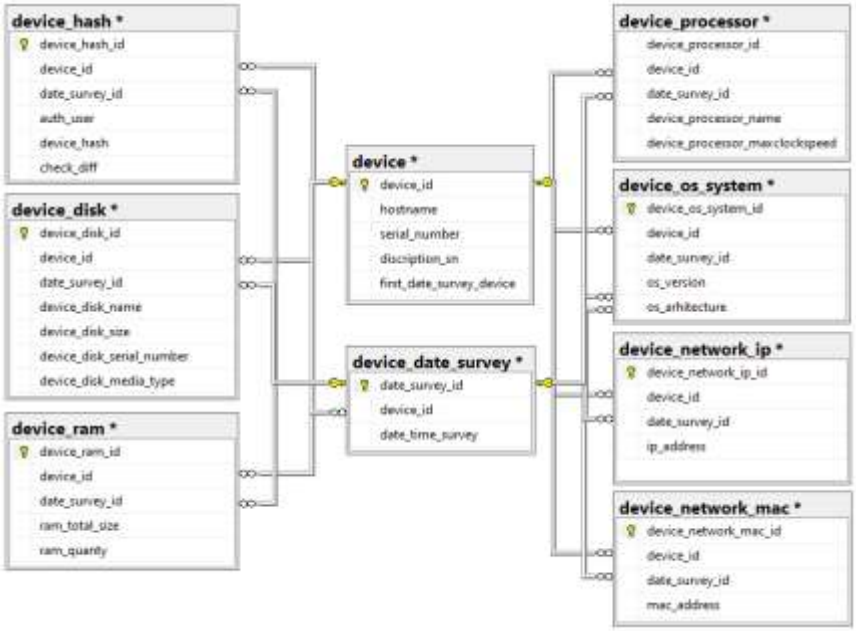

Рис. 2. Диаграмма MSSQL базы данных

2. Программная реализация

Использованный для программной реализации скриптовый язык PowerShell является наиболее подходящим средством, так как он уже по умолчанию установлен в операционной системе Windows, в нем уже имеются библиотеки для сбора системных параметров и взаимодействия с MSSQL базой данных.

1. Проверка исключений.

Так как пользователи на которых распространяется групповая политика могут авторизоваться на компьютерах из списка исключений, в скрипт необходимо добавить условие прерывания, представленное на рис. 3.

```
if (($(HOSTNAME) -match "доменное имя компьютера из списка исключений") `
    -or ($(HOSTNAME) -match "доменное имя компьютера из списка исключений"))
{
    exit
}
```

Рис. 3. Фрагмент кода прерывания

2. Сбор информации.

Команды:

1) Фрагмент кода сбора информации о имени компьютера представлен на рис. 4.

```
$Device_Hostname = HOSTNAME
```

Рис. 4. Фрагмент кода сбора информации о имени компьютера

2) Фрагмент кода сбора информации о серийном номере компьютера представлен на рис. 5.

```
$Device_serial_number = Get-WmiObject win32_bios | `
    Select-Object -ExpandProperty SerialNumber
```

Рис. 5. Фрагмент кода сбора информации о серийном номере

3) Фрагмент кода сбора информации о модели процессора компьютера представлен на рис. 6.

```
$Device_Processor_name = (Get-WmiObject -Class Win32_Processor).name
```

Рис. 6. Фрагмент кода сбора информации о модели процессора

4) Фрагмент кода сбора информации о максимальной частоте процессора компьютера представлен на рис. 7.

```
$Device_Processor_MaxClockSpeed = (Get-WmiObject -Class Win32_Processor).MaxClockSpeed
```

Рис. 7. Фрагмент кода сбора информации о максимальной частоте процессора

5) Фрагмент кода сбора информации о названии операционной системы компьютера представлен на рис. 8.

```
$Device_OsName = (Get-WmiObject Win32_OperatingSystem).Caption
```

Рис. 8. Фрагмент кода сбора информации о названии операционной системы

6) Фрагмент кода сбора информации об архитектуре операционной системы компьютера представлен на рис. 9.

```
$Device_OsArchitecture = (Get-WmiObject Win32_OperatingSystem).OSArchitecture
```

Рис. 9. Фрагмент кода сбора информации об архитектуре операционной системы

7) Фрагмент кода сбора информации об объеме оперативной памяти компьютера представлен на рис. 10.

```
$Device_RAM = (Get-WmiObject Win32_PhysicalMemory | Measure-Object `
              -Property capacity -Sum).sum /1GB
```

Рис. 10. Фрагмент кода сбора информации об объеме оперативной памяти

8) Фрагмент кода сбора информации о количестве плат оперативной памяти компьютера представлен на рис. 11.

```
$Device_RAM_quantity = (Get-WmiObject Win32_PhysicalMemory | Measure-Object).count
```

Рис. 11. Фрагмент кода сбора информации о количестве плат оперативной памяти

9) Фрагмент кода сбора информации об ip-адресе компьютера представлен на рис. 12.

```
$Device_IP_address = $($(Test-Connection -ComputerName (HOSTNAME) `
                    -Count 1).IPV4Address).IPAddressToString
```

Рис. 12. Фрагмент кода сбора информации об ip-адресе компьютера

10) Фрагмент кода сбора информации об MAC-адресе компьютера представлен на рис. 13.

```
$Device_MAC_address = $($(getmac /fo table /nh)[1]).Substring(0, 17)
```

Рис. 13. Фрагмент кода сбора информации об MAC-адресе компьютера

3. Формирование хэш-записи.

Фрагмент кода формирования хэш-записи представлена на рис. 14.

```
function Hash($textToHash)
{
    $hasher = new-object System.Security.Cryptography.MD5CryptoServiceProvider
    $toHash = [System.Text.Encoding]::UTF8.GetBytes($textToHash)
    $hashByteArray = $hasher.ComputeHash($toHash)
    foreach ($byte in $hashByteArray)
    {
        $res += $byte.ToString()
    }
    return $res;
}
```

Рис. 14. Фрагмент кода формирования хэш-записи

Фрагмент кода обращения к функции формирования хэш-записи представлена на рис. 15.

```
$Device_Hash = $(Hash $($Device_Hostname + #Системные параметры сохраненные в переменные#))
```

Рис. 15. Фрагмент кода обращения к функции формирования хэш-записи

4. Фрагмент кода подключения к MSSQL базе данных представлена на рис. 16.

```
$SqlConnection = New-Object System.Data.SqlClient.SqlConnection
$SqlConnection.ConnectionString = "Server = ip-адрес; Database = название базы ; `
                                  User Id = login; Password = password;"
$SqlCmd = New-Object System.Data.SqlClient.SqlCommand
$SqlCmd.Connection = $SqlConnection
$SqlAdapter = New-Object System.Data.SqlClient.SqlDataAdapter
$DataSet = New-Object System.Data.DataSet
$SqlConnection.Open()
$DataSet.Clear();
```

Рис. 16. Фрагмент кода подключения к MSSQL базе данных

5. Фрагмент кода проверки наличия доменного имени компьютера в MSSQL базе данных представлена на рис. 17.

```
$SqlQuery = "SELECT COUNT(*) FROM device WHERE hostname = '$(HOSTNAME)'"
$SqlCmd.CommandText = $SqlQuery
$SqlCmd.Connection = $SqlConnection
    if ($SqlCmd.ExecuteScalar() -gt 0)
    {
        Write-Host "Доменное имя компьютера найдено"
    }
    else
    {
        Write-Host "Доменное имя компьютера не найдено"
    }
```

Рис. 17. Фрагмент кода проверки наличия доменного имени компьютера в MSSQL базе данных

6. Выборка из MSSQL базы данных.

Команда выборки информации из MSSQL базы данных на примере выборки «device_id» из таблицы «device» представлена на рис. 18.

```
$SqlQuery = "SELECT device_id FROM device WHERE hostname = '$(HOSTNAME)'"
$SqlCmd.CommandText = $SqlQuery
$SqlAdapter.SelectCommand = $SqlCmd
$SqlAdapter.Fill($DataSet)
$device_id = $DataSet.Tables[0] | foreach { $_.device_id };
$DataSet.Clear();
```

Рис. 18. Команда выборки информации из MSSQL базы данных

7. Добавление информации в MSSQL базу данных.

Фрагмент кода добавления информации в MSSQL базу данных на примере добавления системных параметров процессора представлена на рис. 19.

```
$SqlQuery = "INSERT INTO device_processor
        (device_id,date_survey_id,device_processor_name,device_processor_maxclockspeed)
VALUES ('$device_id','$date_survey_id','$ARM_Processor_name','$ARM_Processor_MaxClockSpeed');"
$SqlCmd.CommandText = $SqlQuery
$SqlAdapter.SelectCommand = $SqlCmd
$SqlAdapter.Fill($DataSet)
$DataSet.Clear();
```

Рис. 19. Фрагмент кода добавления информации в MSSQL базу данных

Заключение

Представленная статья описывает разработанный подход к автоматизированной инвентаризации системных параметров компьютеров в Windows домене. Разработанная система, состоящая из PowerShell скрипта и MSSQL базы данных, позволяет эффективно собирать и хранить информацию о компьютерах в сети. Автоматический запуск скрипта по групповым политикам обеспечивает регулярное обновление данных, минимизируя трудозатраты на ручную инвентаризацию.

Ключевым преимуществом предлагаемого решения является использование хэш-записи компьютера для отслеживания изменений, что позволяет оптимизировать работу системы, избегая избыточных запросов к базе данных. Это значительно снижает нагрузку на сервер базы данных и повышает производительность системы в целом.

Данная система позволяет получить детальную информацию о каждой машине, включая такие характеристики, как модель процессора, объем оперативной памяти, установленное ПО, IP и MAC адреса. Значимость этой информации для управления ИТ-инфраструктурой неоспорима, предоставляя возможность оперативно реагировать на изменения, проводить модернизацию и обеспечивать безопасность сети. Использование централизованной базы данных позволяет удобно анализировать и использовать полученную информацию для принятия стратегических решений.

В заключение, представленный подход обеспечивает автоматизированное решение задачи инвентаризации, повышая эффективность управления и безопасности ИТ-инфраструктуры в Windows домене. Дальнейшее развитие может включать интеграцию с другими системами управления, например, с системами мониторинга или Service Desk, для обеспечения еще более комплексного управления ИТ-активами.

Список литературы

1. Программы для инвентаризации оборудования: официальный сайт [Электронный ресурс]. Режим доступа: https://serveradmin.ru/top-10-programm-dlya-inventarizaczii-oborudovaniya/ (дата обращения: 03.12.2024).

An approach to the inventory of system parameters of computers in the Windows domain

***Halikov, I.I.*[1,*] *, Makarova, A.Y.*[1], *Podobrii, A.N.*[2,3], *Pertsev, A.A.*[2,3]**

[*] halikov_ii@mail.ru

[1] Ulyanovsk State University, Russia

[2] FPRC JSC 'RPA 'Mars', Ulyanovsk, Russia

[3] Ulyanovsk State Technical University, Russia

The article presents an approach to automated inventory of system parameters of computers in the Windows domain. The developed solution uses a PowerShell script, run according to group policies, to collect information about the characteristics of computers, such as the processor model, memory capacity, installed operating system, and other parameters. The collected data is stored in the MS SQL database. The implementation of the hash recording mechanism allows tracking changes in computer characteristics and optimizing the frequency of writing data to the database. The proposed approach provides efficient and automated management of the OT infrastructure, significantly reducing labor costs and increasing the efficiency of decision-making. The article describes the architecture of the system, the algorithms of the script, and the structure of the database, providing practical guidance on the implementation of automated inventory.

***Keywords**: inventory of system parameters, Windows domain, PowerShell, MSSQL*