Win11 Setup Batch

Metadata

title: Win11 Setup Batch
name: Win11SetupBatch
description: Win11 Setup Batch
type: overview
categories: business
topics: business
tags:
  - #note

id: d26174
uid: d9249259-22dd-4bf3-8275-6cfcf437022e
date: 2024-08-28T17:58:02
created_at: 1724835482
updated_at: 1724835482
path: Notes/pending/win11-setup-batch.md
slug: win11_setup_batch
url:
https://username.github.io/repo/posts/2024/08/28/0/1/win11-setup-batch
redirect_from:
  -

lang: en
author: undefined
private: true
weight: 1
toc: false
draft: true
status:
keywords:
changelog:
versions:

Abstract

Introduction

Methodologies

Prerequisites

- Windows 11 ISO file
- USB drive (at least 8GB)
- Administrative privileges

Download Windows 11 ISO

```
Invoke-WebRequest -Uri "https://www.microsoft.com/en-us/software-download/windows11" -OutFile "Win11.iso"
```

Create Bootable USB Drive

```
diskpart /s CreateUSB.txt
```

CreateUSB.txt

```
select disk <DiskNumber>
clean
create partition primary
select partition 1
format fs=ntfs quick
active
assign letter=<USBDriveLetter>
exit
```

Mount ISO and Copy Files to USB

```
$iso = "Win11.iso"
$mount = Mount-DiskImage -ImagePath $iso -PassThru | Get-Volume
robocopy $mount.Root "\<USBDriveLetter>\" /s /e
Dismount-DiskImage -ImagePath $iso
```

Configure Unattended Installation

Autounattend.xml

```
<unattend xmlns="urn:schemas-microsoft-com:unattend">
    <settings pass="windowsPE">
        <component name="Microsoft-Windows-Setup" language="neutral">
            <UserData>
                <AcceptEula>true</AcceptEula>
                <FullName>YourName</FullName>
                <Organization>YourOrg</Organization>
                <ProductKey>YourProductKey</ProductKey>
            </UserData>
        </component>
    </settings>
</unattend>
```

Copy Unattended File to USB

```
Copy-Item "Autounattend.xml" "\<USBDriveLetter>\"
```

Automate Driver Installation

```
$drivers = Get-ChildItem -Recurse "C:\Drivers\*.inf"
foreach ($driver in $drivers) {
    pnputil /add-driver $driver.FullName /install
}
```

Automate Software Installation

```
$apps = @("App1", "App2", "App3")
foreach ($app in $apps) {
    Start-Process -FilePath "C:\PathToInstallers\$app.exe" -ArgumentList "/quiet" -Wait
}
```

Automate Windows Updates

```
Install-PackageProvider -Name NuGet -Force
Install-Module PSWindowsUpdate -Force
Get-WindowsUpdate -Install -AcceptAll -AutoReboot
```

Post-Installation Configuration

```
# Disable Cortana
Set-ItemProperty -Path "HKCU:\Software\Microsoft\Personalization\Settings" -Name "SettingsEnabled" -Value 0

# Set Power Plan
powercfg /change monitor-timeout-ac 0
powercfg /change monitor-timeout-dc 0
powercfg /change standby-timeout-ac 0
powercfg /change standby-timeout-dc 0

# Configure Firewall
Set-NetFirewallProfile -Profile Domain,Public,Private -Enabled True
```

Application Setup

Download Application List

```
wget https://example.com/winget-apps.txt -OutFile "winget-apps.txt"
```

Install Applications via Winget

```
winget import -i winget-apps.txt --accept-source-agreements --accept-package-agreements
```

Install Applications via Chocolatey

```
choco install -y 7zip git vscode googlechrome
```

Custom Application Installers

```
$urls = @(
    "https://example.com/app1_installer.exe",
    "https://example.com/app2_installer.msi"
)
foreach ($url in $urls) {
    $file = Split-Path -Leaf $url
    wget $url -OutFile $file
    if ($file.EndsWith(".exe")) {
        Start-Process -FilePath $file -ArgumentList "/quiet" -Wait
    } elseif ($file.EndsWith(".msi")) {
        Start-Process -FilePath "msiexec" -ArgumentList "/i $file /quiet" -Wait
    }
}
```

Security Setup

Enable BitLocker

```
Enable-BitLocker -MountPoint "C:" -RecoveryPasswordProtector -PasswordProtector
```

Enable Windows Defender

```
Set-MpPreference -DisableRealtimeMonitoring $false
Update-MpSignature
Start-MpScan -ScanType FullScan
```

Configure Firewall Rules

```
# Block all incoming connections
Set-NetFirewallProfile -Profile Domain,Public,Private -DefaultInboundAction Block

# Allow specific apps
New-NetFirewallRule -DisplayName "AllowApp1" -Direction Inbound -Program "C:\Program Files\App1\App1.exe" -
Action Allow
New-NetFirewallRule -DisplayName "AllowApp2" -Direction Outbound -Program "C:\Program Files\App2\App2.exe" -
Action Allow
```

Secure Remote Desktop

```
Set-ItemProperty -Path "HKLM:\System\CurrentControlSet\Control\Terminal Server" -Name "fDenyTSConnections" -
Value 0
New-NetFirewallRule -Name "RDP" -Protocol TCP -LocalPort 3389 -Direction Inbound -Action Allow -Profile
Domain,Private
```

Privacy Setup

Disable Telemetry

```
reg add "HKLM\Software\Policies\Microsoft\Windows\DataCollection" /v AllowTelemetry /t REG_DWORD /d 0 /f
```

Disable Advertising ID

```
reg add "HKCU\Software\Microsoft\Windows\CurrentVersion\AdvertisingInfo" /v Enabled /t REG_DWORD /d 0 /f
```

Disable Location Tracking

```
reg add "HKCU\Software\Microsoft\Windows\CurrentVersion\CapabilityAccessManager\ConsentStore\location" /v Value
/t REG_SZ /d Deny /f
```

Disable App Suggestions

```
reg add "HKCU\Software\Microsoft\Windows\CurrentVersion\ContentDeliveryManager" /v SystemPaneSuggestionsEnabled
/t REG_DWORD /d 0 /f
```

Final Configuration

Configure User Account Control (UAC)

```
reg add "HKLM\Software\Microsoft\Windows\CurrentVersion\Policies\System" /v EnableLUA /t REG_DWORD /d 1 /f
```

Set Windows Update Policy

```
reg add "HKLM\Software\Policies\Microsoft\Windows\WindowsUpdate\AU" /v NoAutoUpdate /t REG_DWORD /d 0 /f
reg add "HKLM\Software\Policies\Microsoft\Windows\WindowsUpdate\AU" /v AUOptions /t REG_DWORD /d 4 /f
```

Configure Backup Settings

```
wbadmin enable backup -addtarget:\\server\share -include:C: -schedule:09:00
```

Disable Windows 11 Startup Sound

```
reg add "HKCU\AppEvents\Schemes\Apps\.Default\SystemExit\.Current" /v ExcludeFromCPL /t REG_SZ /d 1 /f
```

Final Reboot

```
shutdown /r /t 0
```

References

1. 1.
    1.