New printing protocols in Samba

Günther Deschner
<gd@samba.org>
Agenda

- MS16-087
- RPRN and PAR
- PAR support detection
- Print Driver Packages
- Driver Signing
- Core Printer Drivers
- Current state of PAR in Samba
- Next steps
- Further reading & Q/A
Samba at RedHat

- Part of Red Hat Gluster Storage Team
- Close relationship with RHEL / Identity Team
- Often collaborate with Andreas Schneider <asn@samba.org> on Samba feature or bugfix development such as printing related matters
MS016-087 (CVE-2016-3238)

- July 2016: Microsoft released security update to address a critical vulnerability in the Windows spooler components
- This update addressed issue by:
  - Correcting Windows spooler access to filesystem
  - Issue a warning when untrusted printer drivers are attempted to be used
- V3 non-package aware printer drivers will get security prompt:
MS016-087 (CVE-2016-3238)

- For non-interactive scenarios, the installation of untrusted printer drivers fails completely (!)
- September 2016: Discussed issue at Samba / Interoperability Lab in Redmond
- October 2016: Microsoft issued follow-up update to mitigate the Point and Print restrictions via white-listing of print servers via Group Policy
- Detailed instructions for this mechanism are both described on
  - support.microsoft.com
  - wiki.samba.org
- What is the real resolution?
MS16-087 resolution:

“Update the affected printer driver. Package-aware V3 printer drivers were introduced in Windows Vista. Installing a package-aware printer driver will resolve the issue.”
Samba needs to support package-aware printer drivers!
What is a package-aware printer driver?

- A package-aware driver typically comes as a driver package
- Microsoft Cabinet Files (.cab)
  - Printer Driver Inf File (.inf)
  - Driver Catalog File (.cat)
  - "Amd64", "x86" directories
- Advantages of Point and Print with driver packages:
  - All runnable components are part of driver package
  - Driver signing and integrity can be checked on the client during installation
  - Easier to manage (less likely to have overlapping driver files)
What is a package-aware printer driver?

- **PackageAware keyword in driver.inf:**

  ;These sections are to identify the Vista drivers as "Package Aware" to allow them to take advantage of features such as "Package Point-and-Print" in Vista and above

  [PrinterPackageInstallation.amd64]
  PackageAware=TRUE
  CoreDriverDependencies={D20EA372-DD35-4950-9ED8-A6335AFE79F0}
What is a package-aware printer driver?

- **Package awareness flag in** `PrinterDriverAttributes` of `PRINTER_INFO_2`
  - `PRINTER_DRIVER_PACKAGE_AWARE = 0x00000001`

- **Accessible in the driver configuration backend, the Windows registry:**
  - `HKLM\System\CurrentControlSet\Control\Print\Enviroments\Windows x64\Drivers\Version-3\DRIVERNAME\PrinterDriverAttributes`

- **People start manipulating this attribute in the registry to pretend these drivers were properly packaged and securely verified**
Can we support package-aware drivers?

- Can we install v3 print driver packages for Point and Print?
- Remember: as Samba does not run off Microsoft OS (usually), we need to let a Windows client prepare everything for Point and Print
- But: spoolss protocol does not provide means to manage package-aware drivers
- => In order to provide package-aware drivers for Point and Print we need to use a different DCE/RPC protocol
For Samba to fully support package-aware printer drivers we need MS-PAR!
RPRN and PAR

- PAR is very similar to RPRN
- PAR inherits the entire messages and marshalling aspects of RPRN
- PAR overcomes fundamental limitations of RPRN
  - Synchronous delivery of printer change notifications
  - Client print server requirement
  - No driver package management capabilities
- 66 PAR calls out of 74 have a 1:1 match to RPRN calls
  - 4 new calls for driver management
  - 4 new calls for change notifications
RPRN and PAR

- **RPRN - “Print System Remote Protocol”**
  - `ncacn_np`
  - “spoolss”
  - available since Windows NT

- **PAR - “Print System Asynchronous Remote Protocol”**
  - `ncacn_ip_tcp`
  - `auth_level >= DCERPC_AUTH_LEVEL_PACKET`
  - use of DCE/RPC header `object_uuid`
    `DCERPC_PFC_FLAG_OBJECT_UUID`
  - “IRemoteWinspool” or “winspool”
  - available since Windows Vista
PAR support detection

- RPRN named pipe is used for PAR detection
- Client calls “OpenPrinterEx” for a print server handle
- Client calls “GetPrinterData” for “OsVersion”
- Client calls “ClosePrinter” for the print server handle
- Client inspects “OsVersion” binary blob
  - MajorVersion
  - MinorVersion
  - BuildNumber
- If “BuildNumber” >= 3791 then PAR will be tried
Printer Change Notifications with RPRN
Printer Change Notifications with PAR
PAR printer change notify

- MS-PAR, 3.1.4.9. Printing Related Notification Methods
  - SyncRegisterForRemoteNotifications
  - SyncUnRegisterForRemoteNotifications
  - SyncRefreshRemoteNotifications
  - AsyncGetRemoteNotifications
Driver upload with “Print Management” and MS-RPRN
Driver upload with MS-RPRN

- Client uploads individual driver files via SMB to print$
- Client calls AddPrinterDriver() DCE/RPC with a fully filled out driver definition
- Server moves files to download area in print$
- Server registers driver definition in backend (registry)
- Server associates driver with printer (if requested)
Driver upload with “Print Management” and MS-PAR
PAR driver management

- MS-PAR, 3.1.4.2. Printer Driver Management Methods
  - AsyncInstallPrinterDriverFromPackage
  - AsyncUploadPrinterDriverPackage
  - AsyncCorePrinterDriverInstalled
  - AsyncDeletePrinterDriverPackage
Driver upload with MS-PAR

- Client uploads driver package components to print$ via SMB
- Client calls AsyncUploadPrinterDriverPackage with SMB path to driver.inf
- Server replies returns driver.inf path of local, private driver repository
- Client calls AsyncInstallPrinterDriverFromPackage with local, private path to driver.inf and driver name
- Server parses driver.inf, creates driver definition, creates driver package cabinet
Driver upload with MS-PAR

- Server has much more work to do with PAR than in RPRN
  - Printer.inf parsing
  - Cabinet creation
Prerequisites for implementing MS-PAR in Samba
DCE/RPC requirements

- Support for ncacn_ip_tcp and endpointmapper
- Support for object_uuid in DCE/RPC header
- Support for DCERPC_AUTH_LEVEL_PACKET
- Thanks to Stefan Metzmacher <metze@samba.org>
Print Driver Package components

- **Microsoft Cabinet Files (.cab)**
  - Well documented format, similar to .tar

- **Printer Driver Inf File (.inf)**
  - Well documented format
  - Driver installation instructions
  - Consumed by Windows Setup API

- **Driver Catalog File (.cat)**
  - Mostly undocumented format
  - Cryptographic signatures of Driver Files

- **“Amd64”, “x86” directories**
  - DLLs, XML files
Cabinet Files – FOSS implementations

- libmspack - https://www.cabextract.org.uk/libmspack/
  - compression support (MSZIP)
  - C library API
  - Client tool (cabextract)
  - NO cabinet creation
  - cabinet extraction

- lcab
  - NO compression
  - NO C library API
  - Client tool (lcab)
  - cabinet creation
  - NO cabinet extraction
Cabinet Files – Samba implementation

- **cab.idl**
  - Samba style standard interface definition
  - autogenerated marshalling code

- **MSZIP compression builtin in libndr**
  - Used for AD replication via DRSUAPI

- **Aurélien Aptel <aaptel@suse.com>**
  - Resolved various issues with MSZIP use in libndr
  - Created new client tool code

- **libcab.so**
  - print server needs to be able to create .cab files on the fly
Driver signing

- Andreas Schneider wrote a .cat file parser “parsemscat”
  - Based on gnutls and libtasn1
  - [https://git.samba.org/?p=asn/samba.git;a=shortlog;h=refs/heads/master
    -mscat](https://git.samba.org/?p=asn/samba.git;a=shortlog;h=refs/heads/master
    -mscat)
  - Needs “Microsoft Root Authority” certificate

- parsemscat allows to fully verify the integrity of files in a printer driver

- PKCS#7 Certificate (Signature) with an embedded data part

- The embedded data is an ASN.1 structure call Certificate Trust List

- It holds checksums (SHA1, SHA256) for files in the driver package
Printer Driver Inf Files

- .inf files can get very complex
- 1 Printer Driver Inf File easily describes 100 printers

Created a parser

- Based on libgpo and Samba internal ini parsing routines
- Creates DRIVER_INFO_8 structure for in-registry store of Drivers

References to Core Printer Drivers

- [PrinterPackageInstallation.amd64]
  PackageAware=True
  CoreDriverDependencies={D20EA372-DD35-4950-9ED8-A6335AFE79F0}
Core Printer Drivers

- Core Printer Drivers provide basic printing components for UNIDRV, PSCRIPT or XPS based printer drivers
- Most Driver Packages refer to Core Printer Drivers
- Core Printer Drivers come with the Windows OS (client and server) and are installed via “Windows Update”
- Core Printer Drivers are not available for public download
- Created “net rpc printer migrate coredrivers” utility to extract Core Printer Drivers directly from a Windows print server
- Defined in every Windows OS in ntprint.inf
Current State of PAR in Samba

- Core Iremotewinspool server included in Samba master
- 90% of Iremotewinspool server calls are detoured to spoolss server automatically (appear as spoolss server calls in the logs)
  - `winspool_AsyncOpenPrinter (0x00) => spoolss_OpenPrinterEx (0x45)`
  - `winspool_AsyncClosePrinter (0x14) => spoolss_ClosePrinter (0x1d)`
- Explicit configuration needed for activation:
  - OsVersion:
    - `spoolss:os_major = 6`
    - `spoolss:os_minor = 1`
    - `spoolss:os_build = 9600`
  - Architecture:
    - `spoolss:architecture = Windows x64`
Current State of PAR in Samba

- spoolsssd setup required
  
  - rpc_server:tcpipt = yes
    rpc_server:epmapper = external
    rpc_server:spoolss = external
    rpc_server:iremotewinspool = external
    rpc_server:register_embedded_np = yes

  - rpc_daemon:epmd = fork
    rpc_daemon:spoolsssd = fork

  - spoolsssd:prefork_max_children = 1
    spoolsssd:prefork_min_children = 1
    spoolsssd:prefork_spawn_rate = 1
Current State of PAR in Samba

- **New local DriverStore location**
  - Currently $LIBDIR/DriverStore/FileRepository
  - Contains extracted driver packages (including Core Drivers)

- **Additional remote Driver Packages store**
  - `\\SERVER\print$\x64\PCC`
  - `\\SERVER\print$\W32X86\PCC`
  - Contains Driver Packages Cabinet Files
Current State of PAR in Samba

- Generously ignoring the new printer change notification calls
  - AsyncGetRemoteNotifications always returns HRES_ERROR_NOT_SUPPORTED

- For running “Print Management” under Windows this seems sufficient for the moment

- We first need to have a fully async DCE/RPC server in Samba in order to support them
Next steps

- **Print Driver Packages**
  - Finish and review Aurelien’s MSZIP work for .cab compression
  - Implement Core Printer Drivers
  - Finish .inf parser

- **Support for v4 Printer Driver model?**

- **DCE/RPC server**
  - Address scaleability of ncacn_ip_tcp:
  - Properly implement association groups
  - Work on asynchronous DCE/RPC
    (long needed for other protocols like witness as well!)

- **Testing**
PAR testing client code in Samba

- rpcclient iremotewinspool command set
- smbtorture
  - rpc.iremotewinspool
  - local.ndr.iremotewinspool
  - local.ndr.spoolss (verifies iremotewinspool RPC packets can be understood as spoolss packets)
PAR Wireshark dissector

- Fully autogenerated dissector based on Samba winspool.idl
- Has no dependency on older spoolss dissector
  - => does not display all structures yet
### PAR Wireshark dissector

#### Wireshark Dissector

**Filter:** iremotewinspool

<table>
<thead>
<tr>
<th>No.</th>
<th>Time</th>
<th>Source</th>
<th>Destination</th>
<th>Protocol</th>
<th>Length</th>
<th>Info</th>
</tr>
</thead>
<tbody>
<tr>
<td>243</td>
<td>16.665797593</td>
<td>192.168.3.40</td>
<td>192.168.3.75</td>
<td>IREMOTEWINSPOOL</td>
<td>322</td>
<td>winspool.AsyncOpenPrinter request</td>
</tr>
<tr>
<td>244</td>
<td>16.667164145</td>
<td>192.168.3.40</td>
<td>192.168.3.75</td>
<td>IREMOTEWINSPOOL</td>
<td>102</td>
<td>winspool.AsyncOpenPrinter response</td>
</tr>
<tr>
<td>252</td>
<td>16.680070793</td>
<td>192.168.3.40</td>
<td>192.168.3.75</td>
<td>IREMOTEWINSPOOL</td>
<td>322</td>
<td>winspool.AsyncClosePrinter response</td>
</tr>
<tr>
<td>254</td>
<td>16.682177112</td>
<td>192.168.3.40</td>
<td>192.168.3.75</td>
<td>IREMOTEWINSPOOL</td>
<td>162</td>
<td>winspool.AsyncClosePrinter response</td>
</tr>
<tr>
<td>257</td>
<td>16.687402099</td>
<td>192.168.3.40</td>
<td>192.168.3.75</td>
<td>IREMOTEWINSPOOL</td>
<td>102</td>
<td>winspool.AsyncClosePrinter response</td>
</tr>
<tr>
<td>261</td>
<td>16.694067262</td>
<td>192.168.3.40</td>
<td>192.168.3.75</td>
<td>IREMOTEWINSPOOL</td>
<td>322</td>
<td>winspool.AsyncClosePrinter response</td>
</tr>
<tr>
<td>265</td>
<td>16.694199090</td>
<td>192.168.3.40</td>
<td>192.168.3.75</td>
<td>IREMOTEWINSPOOL</td>
<td>594</td>
<td>winspool.SyncRegisterForRemoteNotifications request [Long frame (38 bytes)]</td>
</tr>
<tr>
<td>266</td>
<td>16.695747123</td>
<td>192.168.3.40</td>
<td>192.168.3.75</td>
<td>IREMOTEWINSPOOL</td>
<td>162</td>
<td>winspool.SyncRegisterForRemoteNotifications response</td>
</tr>
<tr>
<td>267</td>
<td>16.695902873</td>
<td>192.168.3.40</td>
<td>192.168.3.75</td>
<td>IREMOTEWINSPOOL</td>
<td>162</td>
<td>winspool.AsyncClosePrinter response</td>
</tr>
<tr>
<td>283</td>
<td>16.701539061</td>
<td>192.168.3.40</td>
<td>192.168.3.75</td>
<td>IREMOTEWINSPOOL</td>
<td>322</td>
<td>winspool.AsyncOpenPrinter request</td>
</tr>
<tr>
<td>284</td>
<td>16.705805173</td>
<td>192.168.3.40</td>
<td>192.168.3.75</td>
<td>IREMOTEWINSPOOL</td>
<td>162</td>
<td>winspool.AsyncOpenPrinter request</td>
</tr>
<tr>
<td>285</td>
<td>16.706288138</td>
<td>192.168.3.40</td>
<td>192.168.3.75</td>
<td>IREMOTEWINSPOOL</td>
<td>162</td>
<td>winspool.AsyncOpenPrinter request</td>
</tr>
<tr>
<td>288</td>
<td>16.707322275</td>
<td>192.168.3.40</td>
<td>192.168.3.75</td>
<td>IREMOTEWINSPOOL</td>
<td>530</td>
<td>winspool.SyncRegisterForRemoteNotifications request</td>
</tr>
<tr>
<td>290</td>
<td>16.707376619</td>
<td>192.168.3.40</td>
<td>192.168.3.75</td>
<td>IREMOTEWINSPOOL</td>
<td>322</td>
<td>winspool.AsyncOpenPrinter request</td>
</tr>
<tr>
<td>294</td>
<td>16.707376619</td>
<td>192.168.3.40</td>
<td>192.168.3.75</td>
<td>IREMOTEWINSPOOL</td>
<td>162</td>
<td>winspool.AsyncOpenPrinter request</td>
</tr>
<tr>
<td>299</td>
<td>16.707137661</td>
<td>192.168.3.40</td>
<td>192.168.3.75</td>
<td>IREMOTEWINSPOOL</td>
<td>545</td>
<td>winspool.SyncRefreshRemoteNotifications request [Malformed Packet]</td>
</tr>
<tr>
<td>300</td>
<td>16.707137661</td>
<td>192.168.3.40</td>
<td>192.168.3.75</td>
<td>IREMOTEWINSPOOL</td>
<td>162</td>
<td>winspool.AsyncClosePrinter request</td>
</tr>
<tr>
<td>306</td>
<td>16.709197573</td>
<td>192.168.3.40</td>
<td>192.168.3.75</td>
<td>IREMOTEWINSPOOL</td>
<td>162</td>
<td>winspool.AsyncClosePrinter request</td>
</tr>
<tr>
<td>307</td>
<td>16.709197573</td>
<td>192.168.3.40</td>
<td>192.168.3.75</td>
<td>IREMOTEWINSPOOL</td>
<td>162</td>
<td>winspool.AsyncClosePrinter request</td>
</tr>
</tbody>
</table>

The Wireshark dissector is a tool that allows for the parsing and visualization of network protocol traffic. It is particularly useful for analyzing and troubleshooting network protocols, such as SMB, which is commonly used in the context of file sharing and printing services in a network environment.
Further reading

- Microsoft Protocol Documentation:
  - MS-RPRN, MS-PAR, MS-PRSOD

- Printer Driver INF Files:
  - https://docs.microsoft.com/en-us/windows-hardware/drivers/print/printer-inf-files

- Microsoft Cabinet File Format:
  - https://msdn.microsoft.com/library/bb417343.aspx#cabinet_format

- MS16-087: Security update for Windows print spooler components: July 12, 2016
Questions and answers

- Mail gd@samba.org, asn@samba.org
- #samba-technical on irc.freenode.net
- WIP branches:
  - https://git.samba.org/?p=gd/samba/.git;a=shortlog;h=refs/heads/master-par-ok
  - https://git.samba.org/?p=gd/wireshark/.git;a=shortlog;h=refs/heads/master-iremotewinspool
Thank you for your attention!

www.redhat.com
www.samba.org

<gd@samba.org>