jNQ™: A Pure Java SMB Client

© 2021 Visuality Systems Ltd

Table of Contents

Table of Figures

Executive Summary

jNQ™ is Visuality's latest and most compliant implementation of Microsoft's SMB network file sharing protocol. With jNQ™, Java developers have now access to legacy SMB1, SMB2.x, and the most up to date features of the SMB3.x dialect.

Written in pure Java, jNQ™ is a client software library, compatible with any Java environment starting from 1.5. Using the jNQ™ API, Java developers can access SMB shares and files, and can leverage security features such as message signing, session encryption, pre-logon integrity, active directory identity management, and Kerberos authentication.

In recent years, cyber-attacks such as WannaCry and Petya have accelerated the move away from SMB1 in favor of SMB2 and SMB3 (Microsoft disabled SMB1 by default in Windows 10 RS3), driving the need for an up-to-date SMB implementation in Java.

jNQ is the only SMB over JAVA solution in the market that is shipped with SMB Microsoft Patents under license agreement with Microsoft.

The Lack of an Up-to-Date SMB Solution for Java

SMB is the **S**erver **M**essage **B**lock protocol.

Originally developed by Barry Feigenbaum at IBM in the 1980's, SMB was referred to as CIFS (**C**ommon **I**nternet **F**ile **S**ystem), but has since been renamed to SMB1 and the term "CIFS" has become less popular. Years of experience with the original SMB protocol led to the creation of SMB2, which further evolved into SMB3. jNQ™ supports all three iterations of SMB.

SMB is a Network File Protocol. It allows clients to access remote files over a network as if they were stored on a local drive. Files in a remote SMB "share" can be created, read, written, edited, executed, renamed, and deleted just like local files.

jNQ™ respects all SMB access controls. A system administrator or individual user has control over who may access their files, and how those files may be accessed.

The SMB3 version of the protocol provides a number of enhancements for secure file transfer, performance, scalability, and resilience.

Java is a major platform for network software development, but a Java implementation of SMB that supports the latest specifications has been sorely lacking. Now that SMB1 has been disabled by default on Windows, the need for a Java toolkit that supports the latest SMB2 and SMB3 dialects has become imperative. Support for SMB3.1.1, the latest SMB dialect, is now an industry requirement.

The Solution

Visuality Systems provides Java developers with our latest and most up-to-date implementation of Microsoft's SMB file sharing protocol. jNQ™, written in pure Java, is a client software library with a well-defined and documented API that supports all SMB versions from SMB1 (CIFS) through SMB3.1.1.

jNQ™ was developed based on Visuality's 20+ years of experience with SMB.

jNQ highlights:

- Working under any Java environment (Oracle, OpenJDK, Android, and IBM).
- Single License; no third party nor open source packages included.
- Sensitive data is protected by leveraging the latest signing and encryption techniques enabled by the protocol.
- Access to improved authentication via Active Directory, Kerberos, and pre-logon message integrity is provided.
- New performance features such as Jumbo Frames are supported, allowing for larger (faster) read and write operations.
- Distributed File-Systems (DFS) support.
- SMB1 support can be enabled for compatibility with legacy Windows systems; jNQ™ is compatible with all Windows versions since Windows2000.

Architecture of jNQ™

jNQ™ is a pure Java library and runs on any Java platform from version 1.5 (Android 4.4.x) upwards. This approach provides compatibility with numerous platforms such as Windows, macOS, Linux, UNIX, Android, etc.

Figure 1 provides an overview of jNQ™'s principal classes:

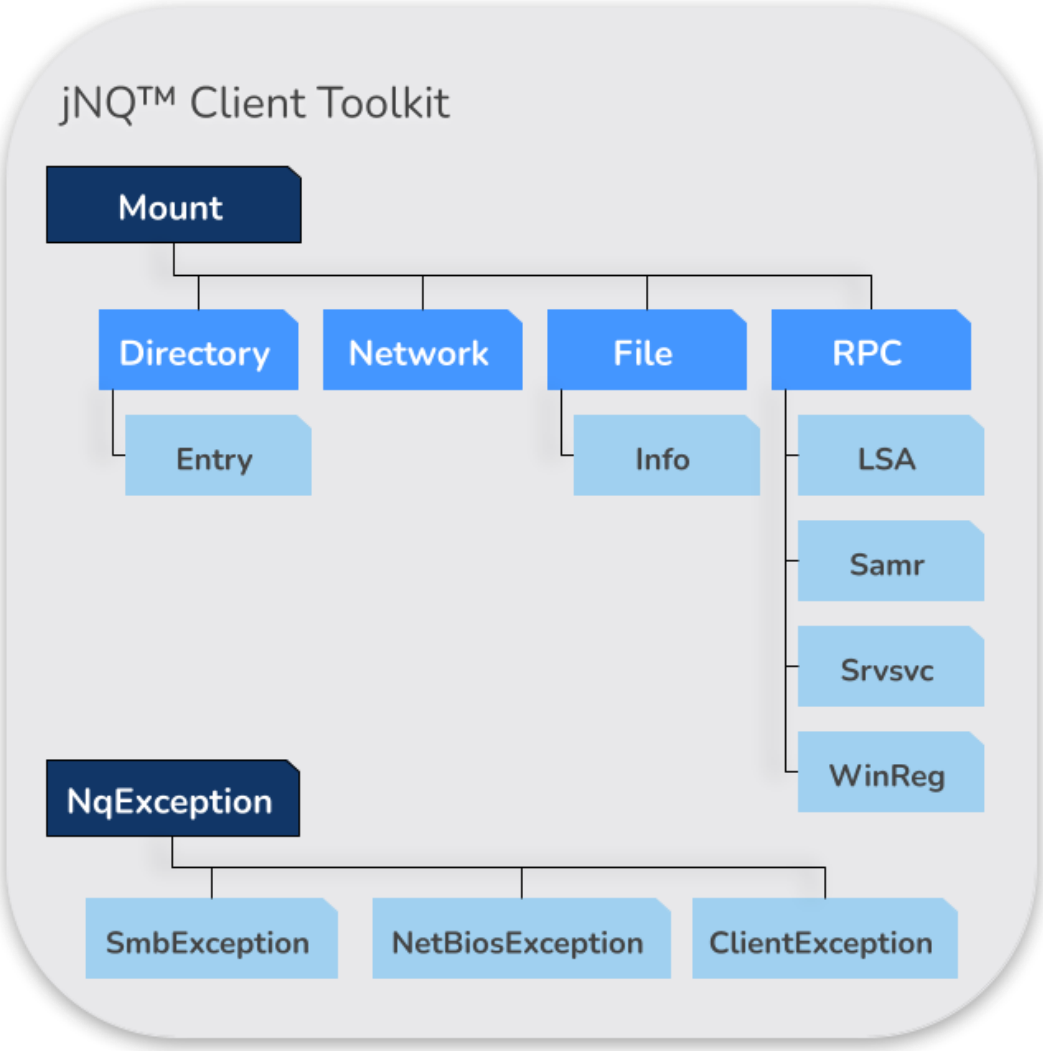

Figure 1: The Architecture of jNQ™

Using jNQ™

jNQ™ provides a robust, well documented API. These are some of the more important classes and methods:

Main classes:

Class	Method examples	Aim
Mount (*)		Establish a connection to a remote share
File (*)	read(), write(), rename(), delete(), getInfo(), setInfo(), close()	Open a file for sync/async reading, writing, or meta-data operations
SmbInputStream SmbOutoutStream		Stream wrappers around file handles
Directory		Directory scan
Network (*)	enumerateDomains(), enumerateServers(), enumerateShares()	Network discovery
Client	checkCredentials(), setDialect()	Client management

RPC classes:

Class	Method examples	Aim
Dcerpc (*)		A generic DCERPC framework for developing yet another RPC
Dssetup (*)	roleGetPrimaryDomainInformation()	DSSETUP sub-pipe of LSA pipe
Lsar (*)	openPolicy(), lookupName(), lookupSid(), queryDomainInfo()	LSA pipe
Samr (*)	openDomain(), lookupDomain(), getmemberNamesInAlias(), openAlias()	SAMR pipe
Srvsvc (*)	shareEnum(), shareAdd(), shareDel(), shareGetInfo(), shareSetInfo()	SRVSVC pipe
Winreg (*)	openLocalMachine(), openKey(), enumValue(), setValue()	WINREG pipe

Auxiliary classes:

Class	Methods examples	Aim
NqException		jNQ-specific error description
SmbException		SMB error description
NetbiosException		Name resolution error description
PasswordCredentials		Credentials container: user, password, domain
SubjectCredentials		A container for a Kerberos ticket
Config		jNQ configuration parameters

This is only a partial listing of the classes and methods made available by jNQ™.

- File, directory, and RPC methods are subject to SMB access controls; an action may only be available to certain users with specified permissions.
- A complete API Javadoc is available upon request.

Sample application

The following code is a sample application built on the jNQ API.

```
import com.visuality.nq.auth.PasswordCredentials;
import com.visuality.nq.client.Mount;
import com.visuality.nq.common.Buffer;
import com.visuality.nq.common.SmbException;
import com.visuality.nq.client.File.Params;

public class Test {
 public static void main( String[] args ) throws Exception {
  try {

   // Create a credentials instance.
   PasswordCredentials creds =
    new PasswordCredentials( "John",
                             "JohnsPassword",
                             "JohnsDomain" );

   // Access the share (Tree Connect).
   Mount mountpoint = new Mount( "SomeServer", "SomeShare", creds );

   // Create a set of open file attributes.
   Params params = new File.Params( File.ACCESS_WRITE,
                                    File.SHARE_FULL,
                                    File.DISPOSITION_OPEN_IF,
                                    false );

   // Create the file.
   File file = new File( mountpoint, "SomeFolder/test.txt", params );

   // Write to the file.
   Buffer writeData = new Buffer( DATASIZE );
   writeData.data = new String( "my data" ).getBytes();
   writeData.dataLen = writeData.data.length;
   file.write( writeData );

   // Close the file.
   file.close();
  } catch( NqException e ) {
   e.printStackTrace();
  }
```

Figure 2: Sample Application

Functionality

jNQ™ SMB features:

- Support for SMB1, and all SMB2/SMB3 dialects up to SMB 3.1.1.
- SMB Message Encryption
- SMB Message Signing
- Various methods of authentication:
  - ➢ From LM to NTLMV2, either “naked” or wrapped into SPNEGO
  - ➢ Kerberos (not supported in Android)
- Distributed File-Systems (DFS) support.
- A rich set of classes and methods:
  - ➢ Full set of file data operations
  - ➢ Full set of file meta-data operations
  - ➢ File tree traversal; directory search
  - ➢ Network discovery
  - ➢ Run-time tuning via configurable parameters
  - ➢ Synchronous and/or asynchronous reads and writes
  - ➢ Hostname resolution through DNS and WINS
- RPC calls: SAMR, LSA, SRVSVC, WINREG
- Multi-threading
- Jumbo frame reads and writes

© 2021 Visuality Systems Ltd

Compliance and Connectivity

jNQ™ implements the most recent dialect of the SMB1 specification (NTLM 0.12), and all dialects of SMB2/SMB3 up to 3.1.1. Applications built with jNQ™ can connect to all Microsoft, Apple's Macintosh, NQ™, Samba and other SMB supported servers.

Figure 2 is a connectivity matrix comparing jNQ™ against several Windows releases. jNQ™ will negotiate the highest SMB dialect supported by the Windows SMB server.

SERVER / CLIENT	Windows 10 RS3	Windows 10 WS 2016	Windows 8.1 WS 2012 R2	Windows 8 WS 2012	Windows 7 WS 2008 R2	Windows Vista WS 2008	Previous Versions
jNQ	SMB 3.1.1	SMB 3.1.1	SMB 3.02	SMB 3.0	SMB 2.1	SMB 2.0	SMB 1.0
Windows 10 RS3	SMB 3.1.1	SMB 3.1.1	SMB 3.02	SMB 3.0	SMB 2.1	SMB 2.0	
Windows 10 WS 2016	SMB 3.1.1	SMB 3.1.1	SMB 3.02	SMB 3.0	SMB 2.1	SMB 2.0	SMB 1.0
Windows 8.1 WS 2012 R2	SMB 3.02	SMB 3.02	SMB 3.02	SMB 3.0	SMB 2.1	SMB 2.0	SMB 1.0
Windows 8 WS 2012	SMB 3.0	SMB 3.0	SMB 3.0	SMB 3.0	SMB 2.1	SMB 2.0	SMB 1.0
Windows 7 WS 2008 R2	SMB 2.1	SMB 2.1	SMB 2.1	SMB 2.1	SMB 2.1	SMB 2.0	SMB 1.0
Windows Vista WS 2008	SMB 2.0	SMB 2.0	SMB 2.0	SMB 2.0	SMB 2.0	SMB 2.0	SMB 1.0
Previous Versions		SMB 1.0	SMB 1.0	SMB 1.0	SMB 1.0	SMB 1.0	SMB 1.0

Figure 3: jNQ™ Connectivity

Use Cases

jNQ™ can be used in any Java environment from J2SE 5 (originally released as 1.5). jNQ™ is also compatible with Android 4.4.x and above.

- **Document Management**
  jNQ™ supports SMB3 encryption to protect critical small and large scale document transfers between client and server.
- **Application Development**
  Access file shares directly from applications without the intervention of the underlying OS, bypassing the need to mount remote filesystems locally.
- **Network Security**
  SMB is a well-established protocol. Authentication and access controls are natively integrated with Active Directory, and can also be integrated with other identity management systems such as FreeIPA, enabling SIEM (Security Information and Event Management) tracking.
- **Mobile Devices**
  Add SMB client capabilities to mobile apps, providing file browsing and access to remote servers.
- **Embedded Systems and IoT**
  Embedded and IoT devices can leverage jNQ™ to send and receive information. Sensor data can be collected and streamed to a file server for storage and analysis. Devices can use SMB shares to reduce or eliminate the need for local storage.

Summary

Built on Visuality's 20+ years of experience in the SMB market, jNQ™ brings SMB client capabilities to any Java platform or application, under a commercial license.

Visuality Systems developed jNQ™ to fill a long-empty gap in the Java market. Java developers can now leverage the latest SMB dialects, based on up-to-date specifications, with comprehensive support from SMB specialists.

© 2021 Visuality Systems Ltd