jNQ:
A Pure Java
SMB Client

A Visuality Systems White Paper

visualitynq.com

Table of Contents

Executive Summary

jNQ is Visuality Systems' Java implementation of the Microsoft's Server Message Block (SMB) network file sharing protocol. jNQ is an SMB client software library, compatible with any Java environment starting from version 1.8.

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

With jNQ, Java developers can conveniently utilize various versions of the SMB protocol, including legacy SMB1, SMB2.x, and the latest features of SMB3.x. However, due to the increasing prevalence of cyber-attacks such as WannaCry and Petya, the abandonment of SMB1 has been hastened. Microsoft took measures to address this security concern by disabling SMB1 by default in Windows 10 RS3 and higher versions, as well as in Azure. This action aimed to protect users and systems from potential vulnerabilities associated with the outdated SMB1 protocol.

jNQ leverages security features end-to-end, including message signing, session encryption, pre-logon integrity, active directory identity management, and Kerberos authentication to enable access to SMB shares and files safely.

jNQ stands out as a Java SMB solution that enforces encryption in an SMB3 environment, mitigating the risks associated with data breaches and unauthorized access. This feature makes jNQ an ideal choice for organizations and developers who prioritize the security and integrity of their SMB communications.

SMB is an intrinsically complex protocol and Microsoft is coming out with newer versions from time to time. Supporting multiple versions and dialects, each with its own set of features and requirements, can be challenging for IT professionals who must ensure that the network and all connected devices are using compatible versions of the protocol. Thanks to its strategic partnership with Microsoft, Visuality Systems is always ahead of the curve supporting the newest SMB dialects aside older versions to allow organizations to implement and maintain SMB with limited knowledge, without the need for you to dig into the protocol particularities.

Finally, jNQ is the sole commercial Java SMB library to offer support for QUIC (Quick UDP Internet Connections), the transport protocol that provides enhanced speed and reliability for network communications.

The Lack of an Updated SMB Solution for Java

SMB is a network file protocol that allows clients to access and orchestrate remote files (also called "shares") over a network as if they were stored on a local drive, according to the access controls defined by the system administrator or by the file owner.

The SMB protocol, originally developed by Barry Feigenbaum at IBM, was initially known as CIFS (Common Internet File System). Since it was renamed to SMB1, the term "CIFS" became less popular. SMB1 evolved into SMB2 and SMB3, introducing enhancements for secure file transfer, performance, scalability, and resilience to network failures. jNQ supports all three SMB versions.

Java is a major platform for network software development, however a Java implementation of SMB that supports the latest specifications has been sorely lacking, probably due to the complexity of the SMB protocol. When SMB1 was disabled by default on Windows, the need for a Java toolkit supporting the latest SMB2 and SMB3 dialects became imperative. The most recent features supported by jNQ include various levels of end-to-end AES encryption and QUIC, the transport protocol that can overcome many of the limitations of TCP, as described in Will QUIC replace TCP?

The Solution

Visuality Systems provides Java developers with the latest and most updated implementation of Microsoft's SMB file sharing protocol. jNQ is an SMB client software library written in pure Java that can be customized according to the customer needs. It provides a well-defined and documented API, which enables support for all versions of SMB, ranging from SMB1 (CIFS) to SMB3.1.1.

More than two and a half decades of experience and a deep understanding of the SMB protocol, based on a solid partnership with Microsoft, have contributed to the creation of a robust and reliable Java software solution.

jNQ highlights:

- **Portability** - works on any Java environment, including Oracle, OpenJDK, Android, IBM.
- **Single license agreement** - jNQ does not incorporate or rely on any external components or libraries that are subject to separate licensing agreements.
- **Security** - by enforcing the latest signing and end-to-end AES encryption techniques introduced by SMB, sensitive data is protected from unauthorized access and breaches.
- **Authentication** - via Active Directory, Kerberos, and pre-logon message integrity, to safeguard against unauthorized access and ID threats.
- **High performance** - large read/write packets processing to boost operations

- **Distributed File-System support** - DFS facilitates synchronization and sharing of files across multiple servers, enhancing data availability and accessibility.
- **Backward compatibility** - jNQ is compatible with all Windows and SMB versions from Windows 95 onwards.
- **Comprehensive testing framework** - designed to simulate real-world scenarios and validate functionality, performance, security, and reliability.
- **QUIC add-on** - the QUIC transport protocol, supported by jNQ as an add-on, overcomes many of the limitations of TCP, as described in Will QUIC replace TCP?

jNQ Architecture

jNQ is a pure Java library and runs on any Java platform from version 1.8 upwards, making it compatible with any operating system like Windows, macOS, Linux, UNIX, Android, etc.

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

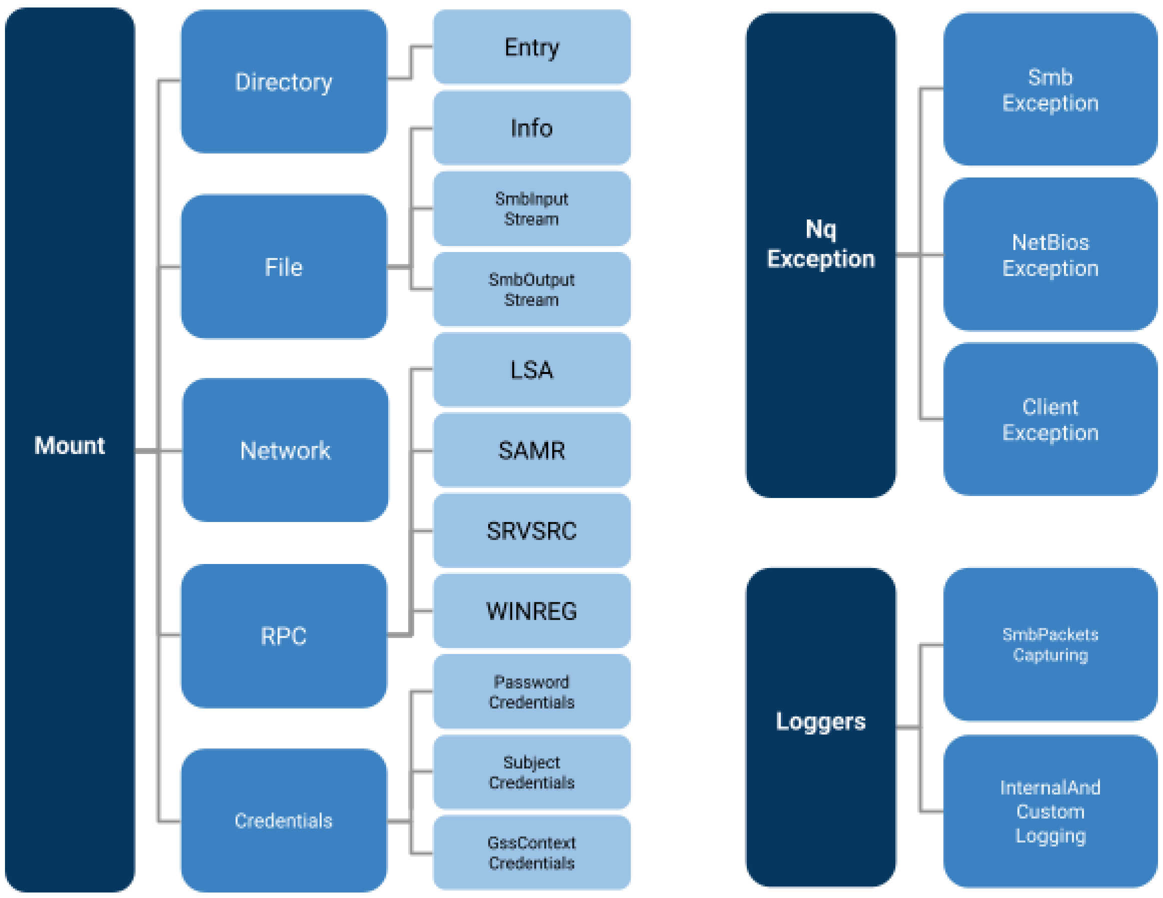

Figure 1: The Architecture of jNQ

jNQ Classes & Methods

jNQ provides a robust, well documented API. Following are some of the most 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
Lsa (*)	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	Method 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

The above is only a partial list of classes and methods available in jNQ.

- File, directory, and RPC methods are subject to SMB access controls; an action may only be available to users with the required permissions.
- A complete API Javadoc is available. Start your free trial now to gain access to the whole documentation and JAR.
- Sample programs are available that demonstrate each of the many jNQ features.

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 {
    int DATASIZE = 1024 * 1024;

    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();

      // Close the SMB connection.
      mountpoint.close();
    } catch (NqException e) {
      e.printStackTrace();
    }
  }
}
```

Features

- Multi-dialect support from SMB1 to SMB 3.1.1
- Authentication - NTLM, Kerberos
- Message signing (AES-based for SMB3)
- Encryption (AES-128-GCM, AES-128-CCM)
- RPC over SMB - Basic – SRVSVC, WINREG
- RPC over SMB - LSA, SAMR
- IPv4 and IPv6 support
- Calls - File data operations, full set of file meta-data, run-time fine-tuning
- Synchronous and asynchronous reads and writes
- Host to IP Resolution via DNS & LLMNR, NetBIOS & WINS
- Network discovery - domain/server/share enumeration (including WS-Discovery)
- Multi-threading
- Durability
- SMB over QUIC with transport-level encryption (add-on)
- Distributed File System (DFS)
- Directory crawling
- Large read/write packets to boost performance
- Server side copy
- Symbolic links

Beyond these features, jNQ can also be tailored to meet customer needs..

Compliance and Connectivity

jNQ implements the most recent dialect of the SMB1 specification (NT LM 0.12), and all dialects of SMB2/SMB3 up to 3.1.1. Applications built with jNQ can connect to any SMB server running on Microsoft Windows, MS Azure, Apple's Macintosh, YNQ, Samba, cloud environments and more.

jNQ will always negotiate the highest SMB dialect supported by a Windows server. The following table shows what's the most recent SMB dialect supported by each Windows version.

Windows	Windows 11 Server 2022	Windows 10 Server 2019	Windows 10 Server 2016	Windows 8.1 Server 2012 R2	Windows 8 Server 2012	Windows 7 Server 2008 R2	Windows Vista Server 2008	Older
SMB Version	3.1.1	3.1.1	3.1.1	3.0.2	3	2.1	2	1

Use Cases

- **Managed File Transfer (MFT)** - Connect to Microsoft Azure and other cloud environments via SMB without any limitations, including SMB3 with encryption and Kerberos authentication over the internet. Easily support SMB and DFS (Distributed File System) dialects and updates.
- **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.
- **Data Discovery & Visibility** - jNQ can be used as a Java-based SMB connector in data discovery and data visibility solutions, enabling users to explore and access data stored in SMB servers and repositories. Developers can leverage the jNQ library to establish secure connections and perform searches.

Summary

Built on Visuality Systems' multi-decade experience in the SMB market, jNQ brings SMB client file sharing capabilities, including encryption, to any Java platform or application, under a commercial license.

Visuality Systems has developed jNQ to fill a gap in the Java SMB arena. Java developers can now leverage the latest SMB dialects, based on up-to-date specifications, with comprehensive 24/7 support from SMB specialists.

Having an SMB protocol challenge?

Drop an email to info@visualitynq.com