Windows Search Protocol recap & update

Noel Power
noel.power@suse.com

Agenda

- Windows Search Protocol recap
- WSP server implementation
  - Recap
  - Implementation problems
  - Solutions
- What's new

Overview

“Windows Search is a desktop search platform that has instant search capabilities for most common file and data types such as email, contacts, calendar appointments, documents, photos, multimedia etc. These capabilities enable users to find, manage, and organize the increasing amount of data common in home and enterprise environments.” - MSDN

Windows Search Service (WSS)

- Builds an index (from a selected location(s)) of a collection of documents by
  - Analyzing files
  - Extracting content, properties & meta data
- Maintains a single index shared by all users
- Maintains security restrictions on content access

Windows Search Protocol

- Allows a client to issue queries to a server hosting the Windows Search service.
- The protocol is primarily intended to be used for full-text queries.
- Uses SMB pipe protocol
- Has a dedicated pipe \pipe\MSFTEWDS allocated for this protocol

WSP server implementation

- Configured as a samba external daemon
- Uses IDL to represent the structures used in the messages defined by the protocol
- Receives WSP queries from a client and converts them into gnome-tracker queries, calls gnome-tracker and converts the results to be passed back to the client

Server Implementation – some problems

Server Implementation – some problems

- Row/Cursor navigation

Server Implementation – some problems

- Row/Cursor navigation
- Row filtering

Server Implementation – some problems

- Row/Cursor navigation
- Row filtering
- Missing Restriction/Property support

Server Implementation – some problems

- Row/Cursor navigation
- Row filtering
- Missing Restriction/Property support
- Testability and lack of client

Server Implementation – some problems

- Row/Cursor navigation
- Row filtering
- Missing Restriction/Property support
- Testability and lack of client support
- Tracker integration

Row/Cursor navigation

- Problem
  - Semantics of WSP protocol demand rows returned for a query are navigable with a cursor.
  - With a GUI for example that cursor is more or less random access as you can scroll anywhere in a list of results.
  - Tracker only offers a cursor than can only be

Row filtering

- Problem(s)
  - WSP protocol doc stipulates that when returning results you need to filter out results such that every row returned by the GSS MUST be ACL checked and rows that don't have file system access to the file must be dropped.
  - Global tracker service which needs of course to

Row filtering

- Solution
  - All rows to be cached returned from tracker are filtered.
  - Filtering done by attempting to open the file (as authenticated user of wsp pipe) for reading, failure open results in row being dropped
- Large results from the query incur performance

Missing Restriction/Property Support

- Problem
  - There are many properties that exist for files/objects, the protocol document only mentions a subset of them. We only support a subset of that subset mentioned in the protocol document.
  - Mapping of properties can be problematic

Missing Restriction/Property Support

- Solution
  - Support for some more properties added
  - Improved logic around best effort query generation
  - Per share enable/disable of WSP to allow piecemeal integration

Testability and lack of client support

- Problem
  - No easy way to test the linux wsp daemon except via windows client
- Windows client is
  - noisy
  - creates complex querys

Testability and lack of client support

- Solution
  - Create a simple cmdline client!

Ways to specify query

- Advance Query Syntax (AQS)
  - “search phrase(s)" AND System.Author:(npower OR noel) AND System.ItemFolderNameDisplay:C:”\MyDocs”
- Windows Search SQL
  - "SELECT Path FROM UserA-4.SystemIndex.Scope() WHERE "SCOPE"=

WSPSEARCH query syntax

- Chose to support Advanced Query Syntax (AQS) or more correctly AQS-like syntax in the cli client because;
  - AQS was best documented see https://msdn.microsoft.com/en-us/library/windows/desktop/bb266512(v=vs.85).aspx#scope

Restriction	Description	Supported
CNodeRestriction	An array of command tree restriction nodes for constraining the results of a query	Y
CContentRestriction	Contains a word or phrase to match for a specific property	Y
CPropertyRestriction	Contains a property to get from each row, a comparison operator and a constant	Y
CNatLanguageRestriction	Contains a natural language query match for a property. Natural language simply means that the string has no formal meaning. The GSS is free to match on the string in a variety of ways. It can drop words, add alternate forms, or make no changes.	Y
CReusewhere	The restriction packet contains a WHEREID that refers to the restriction array used to construct a currently open query	Y
CCoercionRestriction	Contains the modifier and rank coercion operation	N
CProbRestriction	Contains parameters for probabilistic ranking.	N
CVectorRestriction	Contains a weighted OR operation over restriction nodes. Vector restrictions represent queries using the full text vector space model of ranking (see [SALTON] for details).	N
CScopeRestriction	Restricts the files to be returned to those with a path that matches the restriction	N
CInternalPropertyRestriction	Contains a property value to match with an operation.	N
CFeedbackRestriction	Contains the number of relevant documents and a property specification for a relevance feedback query.	N
CRelDocRestriction	Contains a relevant document ID.	N

WSPSEARCH query syntax

- How do we deal with selecting columns to be returned with the query
  - New optional ‘SELECT’ statement

Enum	Keyword
System.ItemDate:System.StructuredQueryType.DateTime#Today	today
System.ItemDate:System.StructuredQueryType.DateTime#Yesterday	yesterday
System.Size#Empty	empty
System.Size#Tiny	tiny

- Example “SELECT System.ItemName, System.ItemURL, System.Size WHERE ALL:$<p403 OR ALL:$<p404 AND System.Kind:picture AND

WSPSEARCH

- Simple command line tool to search remote server using WSP
  - Search for different types e.g. [Calendar|Communication|Contact|Document|Email|Feed|Folder|Game|InstantMessage|Journal|Link|Movie|Music|Note|Picture|Program|RecordedTV|SearchFolder|Task|Video|WebHistory]

Tracker integration

- The Problem
- Tracker is accessed via a glib library; how can we call asynchronous glib api(s) within a tevent based application. To use such api(s) we need to hand over control to the glib main event loop while we wait for a response. We can't of course do that if the application already has and continues to need to hand off to a different main

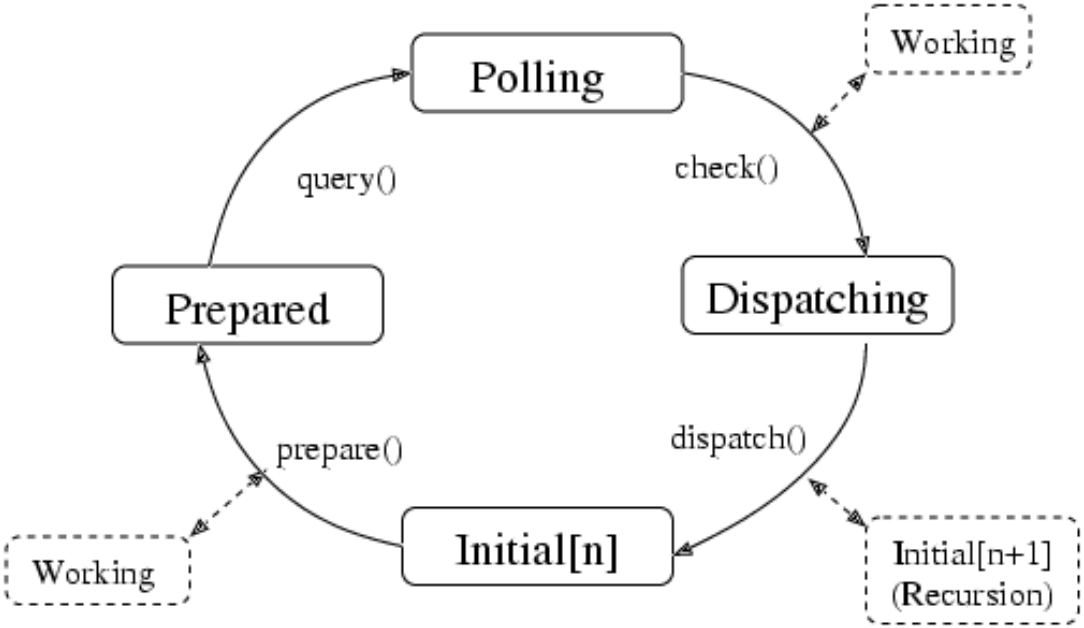
Event loop integration: Parallel

- Advantages
  - Level of separation is easy to define
    - Child process
    - Thread
    - Stand alone process
- Disadvantages

Event loop integration: Master/Slave

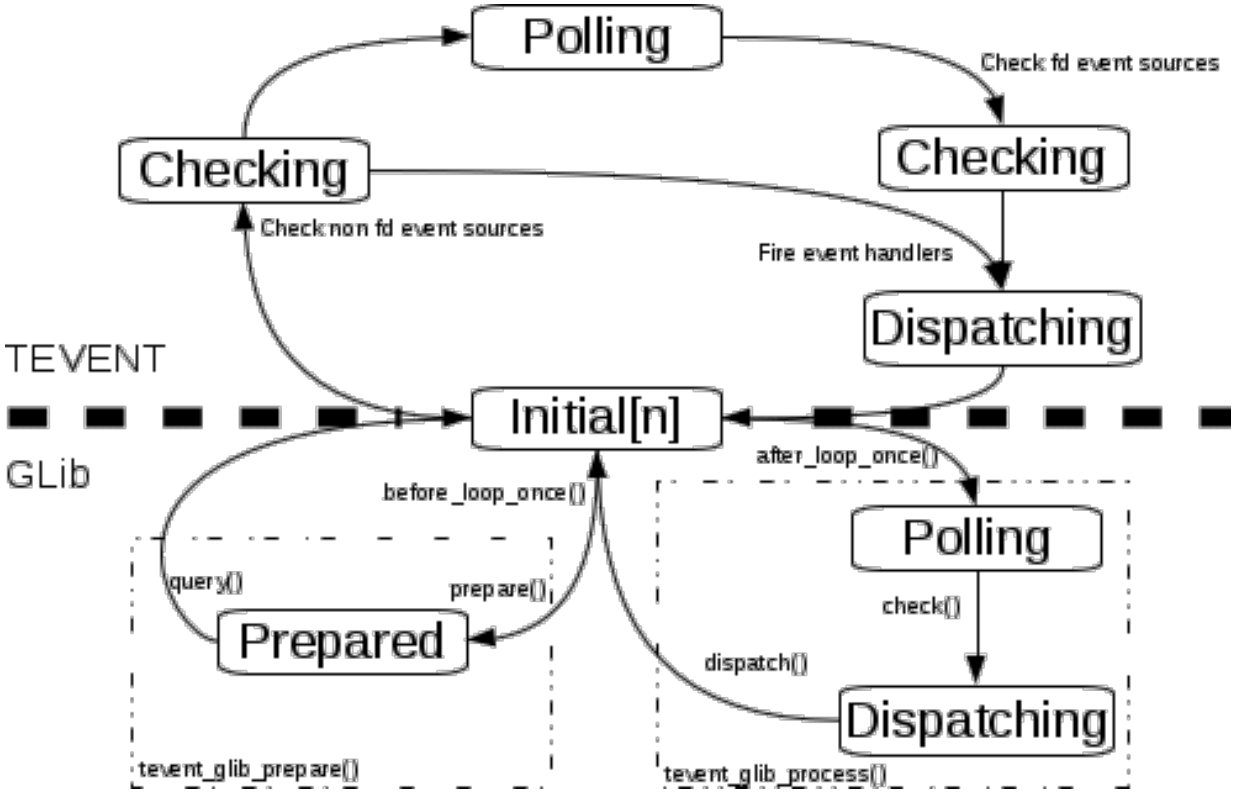
- Advantages
  - Seamless ability to call either asynchronous api from either tevent or glib api(s)
  - No need for bridge code
  - No need to manage other resources
- Disadvantages

GMainContext



https://developer.gnome.org/glib/2.52/mainloop-states.gif

Glueing it all together



Whats new then

- wspsearch – a command-line client to allow dynamic queries to be specified and run against remote servers
- Wireshark dissector – since Wireshark version 1.99.9 Be careful!! this got broken soon after I only noticed that recently so you need to run at least version 2.2.4

How to Help

- The most important!!, please help get this upstream by
  - Review
  - Testing
  - Download the code (details to follow)
- Fix my dodgy code!!

Where can I find the code?

- WSP-WIP
  - This is the most up to date branch, it contains all the changes, it has both client and server implementation code.
    - https://github.com/noelpower/samba/tree/WSP-WIP
- WSP-WIP-NO_RAWPIPE

How do I test it out?

- Option 1:
  - Follow Ralphs steps for setting up Tracker for spotlight, you need to do exactly the same as described here https://wiki.samba.org/index.php/Spotlight#Setup .
- Option 2:

How do I test it out?

- Option 2: (contd.)
  - Enable the services
    - systemctl enable system-tracker-store
    - systemctl enable system-tracker-extract
    - systemctl enable system-tracker-miner-fs
  - Start the services

Questions?


SUSE
We adapt. You succeed.