March 2018

# Forensic String Searching Tool Test Assertions and Test Plan

Public Draft 1 of Version 1.0

National Institute of Standards and Technology
U.S. Department of Commerce

# Abstract

This paper defines test assertions and a test plan for testing digital forensic string search (DFSS) tools used in computer forensics investigations. These test assertions are derived from *Forensic String Searching Tool Requirements Specification, Version 1.0*. The test plan can be used to determine whether a specific tool meets the requirements. The test assertions describe specific statements of conditions that can be checked after a test is executed.  Each assertion generates one or more test cases consisting of a test protocol and the expected test results.  The test protocol specifies detailed procedures for setting up the test, executing the test, and measuring the test results.

As this document evolves updated versions will be posted at http://www.cftt.nist.gov

## TABLE OF CONTENTS

# 1. Introduction

There is a critical need in the law enforcement community to ensure the reliability of computer forensic tools. A means is required to ensure that forensic tools consistently produce accurate, repeatable and objective test results. The goal of the Computer Forensic Tool Testing project at the National Institute of Standards and Technology is to establish a methodology for testing computer forensic tools by development of general tool specifications, test procedures, test criteria, test sets, and test hardware. The results of this working methodology provide helpful information toolmakers can use to improve their tools, so that users of these tools can make informed choices about acquiring and using computer forensic tools, and for interested parties to better understand a tools given capabilities. Our approach for testing computer forensic tools is based on well-recognized international methodologies for conformance testing and quality testing. This project is further described at: http://www.cftt.nist.gov/.

The Computer Forensics Tool Testing (CFTT) program is a joint project of the Department of Homeland Security (DHS), the National Institute of Justice (NIJ), and the National Institute of Standards and Technology (NIST) Special Programs Office and Information Technology Laboratory (ITL). CFTT is supported by other organizations, including the Federal Bureau of Investigation, the U.S. Department of Defense Cyber Crime Center, U.S. Internal Revenue Service Criminal Investigation Division Electronic Crimes Program, and the U.S. Department of Homeland Security's Bureau of Immigration and Customs Enforcement, U.S. Customs and Border Protection and U.S. Secret Service. The objective of the CFTT program is to provide measurable assurance to practitioners, researchers, and other applicable users that the tools used in computer forensics investigations provide accurate results. Accomplishing this requires the development of specifications and test methods for computer forensics tools and subsequent testing of specific tools against those specifications.

Test results provide the information necessary for developers to improve tools, users to make informed choices, and the legal community and others to understand the tools' capabilities. The CFTT approach to testing computer forensics tools is based on well-recognized methodologies for conformance and quality testing. Interested parties in the computer forensics community can review and comment on the specifications and test methods posted on the CFTT Web site (http://www.cftt.nist.gov/). As this document evolves updated versions will be posted at the web site.

# 2. Purpose

This paper defines test assertions and a test plan for testing digital forensic string search (DFSS) tools, often imbedded within a multi-function digital forensic tool, used in computer forensics investigations. These test assertions are derived from *Forensic String Searching Tool Requirements Specification*, *Version 1.0*. The test plan can be used to determine whether a specific tool meets the requirements. The test assertions describe specific statements of conditions that can be checked after a test is executed. Each assertion generates one or more test cases consisting of a test protocol and the expected test results. The test protocol specifies detailed procedures for setting up the test, executing the test, and measuring the test results.

# 3. Scope

The scope of this specification is limited to software tools that search acquired image files created from digital storage media.

# 4. Background

A general model for operation of a DFSS tool is helpful for writing tool requirements and identifying terms useful for discussing string searching. This abstract framework gives structure and provides a general outline with which to proceed during the analysis. At an abstract level string searching involves the following:

- something to search with,
- someplace to search,
- something to search for, and
- search results.

A *search engine* implements a *search algorithm* that finds where a given text string is located within something such as an image file or a storage device. A *digital forensic string search* (DFSS) *tool* provides an interface between a user and a search engine. The DFSS tool interfaces to at least one search engine, but may interface to additional search engines.

Some place must be accessible to the DFSS tool for searching. This place is called the *search universe*. The actual search may be restricted to a subset of the search universe. Usually a forensic tool operates on a set of image files acquired from storage devices obtained in an investigation. An image file may represent an entire storage device divided into several partitions or just a single partition. Such an image file may contain currently active files, deleted files and unallocated space. A search may be initiated for text strings that might identify files relevant to an investigation. In some situations, it may also be desirable to search unallocated space for deleted or hidden information.

In the simplest case, the user is looking for a match from the search universe to a target search string. The tool can search for a key word like *gun* or *knife,* but it might be directed by the user to search for both. In general, the user has a case specific list of search terms. In another case, if the user wants the tool to find social security numbers, groups of nine digits can be specified as a regular expression (i.e., a pattern) such as **[0-9]{9,9}** (a string of nine digits with no separators). In other cases the user might need to search for text that is not represented in ASCII, such as searching for the Chinese word

虎 (*hu* or tiger).  There are multiple possible encodings for the character (e.g., Unicode, GB, Big 5, SHIFT JIS, etc.). It should be noted that English text may also use multiple encodings, e.g., old Univac series computers used an encoding called *field data* and some IBM systems used *extended binary coded decimal interchange code* (EBCDIC) to represent text. The encoding for *Z* in ASCII is 01011010, in EBCDIC it is 11101001 and in field data (six bits) is 011111.

As a practical matter, the *something to search for* is not just a search string but includes a collection of parameters.

After a search is performed the results must be presented to the user in a meaningful and useful way. The actual strings matched, in the case of a pattern search, and the location of the matched strings must be presented in the response such that the matched strings and surrounding context can be extracted for analysis and reporting.

## 5. Definitions

The following terms are used in this document to describe string searching.

| Term | Definition |
|---|---|
| Digital forensic search tool: | Interfaces to one or more *string search engines*. |
| DFSS: | Abbreviation for *digital forensic search tool.* |
| String search engine: | Implements a string search algorithm that takes a *query* and a *search universe* and returns a *query response set.* |
| Query: | A set of *search parameters* that specify a *match set.* |
| Search parameters[*]: | • Search pattern: a string specifying in some *pattern matching language* substrings of the search universe.<br>• Search engine: the implementation of a particular search algorithm that executes a search query.<br>• Character representation: the interpretation of the bit patterns of the search area by the query.<br>• Ignore case: the upper-case and lower-case variations of a character match.<br>• Text direction: the direction words (left-to-right or right-to-left) are written in the text.<br>• Search type: One of *pattern match*, *word*, *stem, physical, logical*, or *index*. These are the most common, but a search engine may define other search types. |
| Pattern matching language: | A language, such as the regular expression language of UNIX used by the **grep** tool, for specifying strings that satisfy a query. |

[*] These are the most common search parameters, but a search engine may define others.

| Term | Definition |
|---|---|
| Location Description: | A location description is composed of four items:<br>1. Matching string<br>2. Object identification (e.g., file name and path)<br>3. Offset within the object (e.g., sector address)<br>4. Length of matching string<br>The offset and length are sometimes omitted. |
| Search hit: | The string matching the query and a *location description* within the search area. |
| Match set: | The set of substrings from the search universe specified by a query, i.e., the expected result to be returned by executing a query. |
| Query response set: | The set of search hits returned by a query. |
| Search universe: | The search universe may be either the content of some type of digital media or an image file taken from some type of digital media. The media may be either unformatted or formatted with one or more file systems (e.g., FAT, NTFS, HFS, etc.). |

# 6. Test Assertions

This section lists test assertions for string searching. The test assertions are described in typical set terminology. A *set* is a collection of *elements*. In this case, a set is a collection of strings. The response set is the set of strings actually returned by a query. The match set is the expected result to be returned by executing a query. An object can be a file, a location in meta-data or a location in unallocated space.

**SS-CA-01.** All elements of the response set are members of the match set for the query.

**SS-CA-02.** All members of the match set are included in the response set for the query.

**SS-CA-03.** All objects containing an element of the response set are identified.

**SS-CA-04.** An accurate location description is included for each element in the response set.

**SS-CA-05.** Text of response displays the appropriate glyph for the text representation.

# 7. Test Data Creation

A set of two image files serves as test data for execution of these test cases. Each image is created with known content and a set of corresponding queries for each test. Each test image covers a set of related file systems. The two images cover the following:

- Windows based file systems: FAT, ExFAT, NTFS and unallocated storage.

- UNIX based file systems: HFS+, HFS+ (Case Sensitive), and ext4.

The following guidelines were followed for image creation:

- Before any partitions or test data are placed on the source drive, a background value of hex zeros is written to each sector.
- Full drive images are created with multiple partitions. Each source drive has multiple partitions.
- The target string of a query is placed in at least one active plain text file and one deleted plain text file in every partition.
- A target string may be represented in ASCII, Unicode or both.
- When Unicode is used, the target string is represented in UTF-8, UTF-16-BE and UTF-16-LE.
  Unicode strings are included to cover the following:
  - Latin based alphabets with diacritical marks, such as Spanish, French and German
  - Right-to-left languages (RTL) such as Hebrew, Arabic, Farsi or Urdu.
  - and Asian languages such as, Chinese, Japanese and Korean (CJK).
  - File names for meta-data search.
- The following special case search strings are created:
  - Search target is in formatted text such as in Microsoft Word or HTML,
  - Search target split across cluster boundary of a fragmented file,
  - Search target located in a location not normally accessible through the operating system and file system,
  - Search target located in a small file stored internally to the NTFS $MFT,
  - Search target embedded in a file name (located in file system meta-data),
  - Search target appears in inflected forms for *stemming* search, e.g., knife and knives.

Additional test images can be created by the tester to cover other areas of interest.

# 8. Test Cases

Each test case is described below.

## 8.1 FT-SS-01: Search ASCII

Search for an ASCII string. The string should be a single word, e.g., *DireWolf*.

## 8.2 FT-SS-02: Search Ignore Case

Search for an ASCII string with *ignore case* turned on. The string should appear in multiple files with variations of case in each file, e.g., *WOLF, Wolf, wolf and DireWolf.*

## 8.3 FT-SS-03: Search for Words

Search for an ASCII string that is a word with *match case* turned on. The target string should match a substring of a second string, but the tested tool should not report a match.

## *8.4 FT-SS-04: Search Logical AND*

Search for a file that contains two target strings, e.g., a file with both the string *panda* and the string *fox*.

## *8.5 FT-SS-05: Search Logical OR*

Search for a file that contains either of two search strings, e.g., either the string *WereWolf* or the string *DireWolf*.

## *8.6 FT-SS-06: Search Logical NOT*

Search for a file that contains one string, but not another given string, e.g., file contains the string *fox*, but the file does not contain the string *tiger*.

## *8.7 FT-SS-07: Search Unicode Text*

All the variations of FT-SS-07 are Unicode string searches for non-English strings. Each variation tests a different situation: CJK (Chinese/Japanese/Korean), non-Latin alphabet, Latin alphabet with diacritical marks, and right-to-left presentation.

### 8.7.1 FT-SS-07-CJK-char: Chinese/Japanese ideograms (Asian)

FT-07-CJK-char tests searching strings from the CJK logographic characters used in Asia, e.g., 中国 (Zhong Guo = China), 東京 (Tokyo).

### 8.7.2 FT-SS-07-CJK-hangul: CJK Korean Hangul (Asian)

Search for a Korean Hangul character, e.g., 서울 (Seoul).

### 8.7.3 FT-SS-07-CJK-kana: CJK Japanese phonetic Kana (Asian)

Search for a string in each of the Japanese syllabic scripts, katakana and hiragana, e.g., スバル (Subaru) and みつびし (Mitsubishi).

### 8.7.4 FT-SS-07-Cyrillic: Non-Latin Cyrillic (Russian)

Search for a target written in Cyrillic, e.g., Сибирь (Siberia).

### 8.7.5 FT-SS-07-Latin: Latin (French & German)

Search for a target containing Latin characters with diacritic marks such as umlauts, accents, tildes or cedillas.

### 8.7.6 FT-SS-07-RTL: Right-To-Left (Arabic)

Search for a target in a language written right-to-left, such as Hebrew, Arabic, Farsi or Urdu, e.g., الكسكس (couscous).

## *8.8 FT-SS-08: Search Tool-defined Queries*

Many tools offer pre-defined queries for common data elements such as email addresses, URLs, IP numbers, credit card numbers, telephone numbers or social security numbers.

### 8.8.1 FT-SS-08-Email: Search Tool-defined Queries -- Email Address

Search for email addresses with a predefined query.

### 8.8.2 FT-SS-08-Phone: Search Tool-defined Queries -- Telephone Number

Search for telephone numbers with a predefined query. The phone numbers should be in a variety of formats such as, (901)555-555, 800-555-5555 or 301.555-5555.

### 8.8.3 FT-SS-08-SS: Search Tool-defined Queries -- Social Security

Search for social security numbers with a predefined query.

## *8.9 FT-SS-09: Special Cases*

There are many unique situations that a string search tool might miss. The variations for test case FT-SS-09 test some common situations that a tool might not be designed to search.

### 8.9.1 FT-SS-09-Doc: Search Formatted Document Text

Formatted documents, such as, Microsoft Office, PDF, or HTML, may imbed formatting tags within a string. Search for targets with embedded formatting.

### 8.9.2 FT-SS-09-Frag: Search Fragmented File

Search for a target that is split across two file fragments. The target will not be found if a tool searches the data image in physical order rather than logical file order.

### 8.9.3 FT-SS-09-Lost: Search Inaccessible (lost) Areas

Search for a target that is placed in a location that is inaccessible to normal file system operations, e.g., in a system area before any partitions, in partition slack after the last cluster, or after the last partition but before the end of the storage device.

### 8.9.4 FT-SS-09-MFT: Search File in MFT

The NTFS file system will store a small file within the Master File Table ($MFT). Search for a string within a file stored in the $MFT.

### 8.9.5 FT-SS-09-Meta: Search file name substring in Meta-data

Search for a target that is a substring of a file name.

### 8.9.6 FT-SS-09-Stem: Search for matches to word stem

Some search tools offer *stemming search*, that is searching for all the inflected forms of a word-stem, not just the word itself. For example, a search for *plan* would also find the following: *plans, planner, planning, planned,* etc. However, *planet* should not be reported as a match.

## *8.10FT-SS-10: Regular Expressions*

A complete test of regular expressions would be a major undertaking, the POSIX specification for regular expressions is over 70 pages. In addition, since forensic tools use several different regular expression grammars, no one set of test cases would work for all tools. The following test cases cover two basic cases.

### 8.10.1 FT-SS-10-Hex: Search Hexadecimal Character Match

Specify the search for an ASCII string as a string of hexadecimal characters, e.g., search for *panda* as `\x70\x61\x6e\x64\x61.`

### 8.10.2 FT-SS-10-Regex: Search Pattern Character Match

Search with a simple regular expression pattern, e.g., [abc] (match either a or b or c).

### 8.10.3 Test Cases Summary

These are the actual search targets for the CFTT String Search test images available from www.cfreds.nist.gov. A test tool is provided with Federated Testing to assist in running the test cases and evaluating the test results.

**Table 1 String Search Test Cases**

| Case | Search Target | Tool Settings | Test Goal |
|---|---|---|---|
| FT-SS-01 | DireWolf | Case = Match Case<br>ASCII = True<br>Unicode = False<br>Whole Words = False | Search ASCII |
| FT-SS-02 | wolf | Case = Ignore Case<br>ASCII = True<br>Unicode = False<br>Whole Words = False | Search Ignore Case |
| FT-SS-03 | Wolf | Case = Match Case<br>ASCII = True<br>Unicode = False<br>Whole Words = True | Search for Words |
| FT-SS-04 | panda and fox | Logical = AND<br>ASCII = True<br>Whole Words = True | Search Logical AND |

| Case | Search Target | Tool Settings | Test Goal |
|---|---|---|---|
| | | Case = Match Case | |
| FT-SS-05 | Were or Dire | Logical = OR<br>Case = Match Case | Search Logical OR |
| FT-SS-06 | fox and not tiger | Logical = NOT<br>Case = Ignore Case | Search Logical NOT |
| FT-SS-07-CJK-char | 中国<br>東京 | Unicode = True | Search Unicode Chinese/Japanese ideograms (Asian) |
| FT-SS-07-CJK-hangul | 서울 | Unicode = True | Search Unicode CJK Korean Hangul (Asian) |
| FT-SS-07-CJK-kana | スバル<br>みつびし | Unicode = True | Search Unicode CJK Japanese phonetic Kana (Asian) |
| FT-SS-07-Cyrillic | Сибирь | Unicode = True | Search Unicode Cyrillic (Russian) |
| FT-SS-07-Latin | garçon<br>Schönheit | Unicode = True | Search Unicode Latin (French & German) |
| FT-SS-07-RTL | الـكسكس | Unicode = True | Search Unicode RTL (Arabic) |
| FT-SS-08-Email | tool defined | Email Address = True<br>Unicode = True | Search Tool-defined Queries -- Email Address |
| FT-SS-08-Phone | tool defined | Telephone Number = True<br>Unicode = True | Search Tool-defined Queries -- Telephone Number |
| FT-SS-08-SS | tool defined | Social Security = True<br>Unicode = True | Search Tool-defined Queries -- Social Security |
| FT-SS-09-Doc | shotgun<br>flintlock<br>rifle<br>revolver<br>longbow<br>crossbow<br>peroxide<br>nitroglycerin | Case = Ignore Case<br>ASCII = True<br>Unicode = True<br>Whole Words = True | Search Formatted Document Text |
| FT-SS-09-Frag | California<br>Washington | Case = Ignore Case<br>ASCII = True<br>Unicode = False<br>Whole Words = True | Search Fragmented File |

| Case | Search Target | Tool Settings | Test Goal |
|---|---|---|---|
| FT-SS-09-Lost | SecretKey<br>Disconnected | Case = Ignore Case<br>ASCII = True<br>Unicode = True<br>Whole Words = False | Search Inaccessable (lost) Areas |
| FT-SS-09-MFT | bear | Case = Ignore Case<br>ASCII = True<br>Unicode = False<br>Whole Words = False | Search File in MFT |
| FT-SS-09-Meta | thunderbird<br>cañón | Case = Ignore Case<br>ASCII = True<br>Unicode = True<br>Whole Words = False | Search file name substring in Meta-data |
| FT-SS-09-Stem | knife<br>steal<br>city<br>plan | Stemming = True<br>ASCII = True<br>Unicode = True | Search for matches to word stem |
| FT-SS-10-Hex | \x70\x61\x6e\x64\x61 | Search Hexadecimal Expression = True | Search Hexadecimal Character Match |
| FT-SS-10-Regex | [DW]..eWolf | Search Regular Expression = True | Search Pattern Character Match |