[linux_forensics] (DATE FIX) announcing bulk_extractor 1.4.0

Simson Garfinkel <simsong@acm.org> Tue, Sep 10, 2013 at 5:17 PM
Reply-To: linux_forensics@yahoogroups.com
To: "linux_forensics@yahoogroups.com" <linux_forensics@yahoogroups.com>

Announcing bulk_extractor 1.4.
Sep 10, 2013

bulk_extractor Version 1.4 has been released for Linux, MacOS and Windows.

It can be downloaded from http://digitalcorpora.org/downloads/bulk_extractor/

New functionality in Version 1.4 includes:

* scan_rar detects RAR archives (both encrypted and unencrypted) and individual RAR archive components (unencrypted). Unencrypted RAR archive components are decompressed and recursively re-processed. Components from encoded (e.g. BASE64, GZIP, etc) RAR files are carved into a special "unrar/" directory. Encrypted RAR files are carved and placed in a special "rar/" directory for additional processing with other tools. Timestamps are preserved when files are carved on Linux or Windows systems.

* scan_exif now implements JPEG carving. By default, only encoded JPEGs are carved (e.g. JPEGs that are gzip'ed or BASE64 encoded, etc). Carved files are created in a directory called "jpeg/". The filename is the forensic path. Confusingly, JPEGs without EXIFs will also be carved.

* scan_zip now implements ZIP component carving. By default, only ZIP components that were encoded are carved. Timestamps are preserved when files are carved on Linux or Windows systems.

* scan_lightgrep will do searches with Lightbox Technology's lightgrep

* scan_hashid will perform hash-based scanning with block hashes.

* scan_xor will search for data hidden with (XOR 255). It is disabled by default, but it is enabled with -e xor or with -e all.

* Variable context window for left and right side.

* Better performance. (Scanners can now declare that they only run at top level; scan_windirs does this. Scanners can also declare that they don't want to receive buffers filled with repeating n-grams or data that they have previously seen; most scanners do this by default.)

* Support for random sampling. (You don't need to scan the entire drive.)

Improvements in existing scanners:

* scan_pdf now does a better job extracting emails from text.

* Files that are carved are now binned in directories of 1000, to
prevent tens of thousands of files from being dumped in a directory.

* scan_accts now writes PII such as SSNs, Fedex#s, and DOBs to the
feature file pii.txt.

=======
Improvements in Python programs:

* Many new features in bulk_extractor_reader.py module for writing programs that parse and process feature files.

* identify_filenames.py now properly reports positions when scan_xor is used.

New features for testing:

* tests/regress.py now performs better validation of reports with the
--validate option. Validation includes making sure that every line is
UTF-8, that feature sizes are less than 1000 characters in size, and
context is less than 1000000 characters in size. We now use --validate
to check the results of BE run against the Real Data Corpus as part of
our release engineering.

* The UTF-8 BOM was removed from the feature files. For a discussion
as to the relative merits and problems with a UTF-8 byte order mark,
see [1].

[1] https://en.wikipedia.org/wiki/Byte_order_mark#UTF-8

Incompatiable changes:

* The -B option for specifying the blocksize for bulk data analysis
has been removed. Instead specify it with -S block_size=NN.

* The -W option no longer specifies the minimum and maximum word
sizes. Instead specify with -S word_min=NN and -S word_max=NN
has been removed. Instead specify it with -S block_size=NN.

==========
OVERREPORTING FIXES

Much of the emphasis of the 1.4 is to decrease "false positives."

* scan_windirs was improperly reporting the cluster for fat directory
entries.

* scan_windirs now only runs at top-level. This means that we will
miss Windows disk images that are GZIP compressed. The advantange
is that it dramatically reduces false positives.

* Additional validations were added to scan_windirs for fat directory
entries. As a result fewer false positives will be reported.

* False positives in scan_net have been dramatically reduced. We have
done this by making the scanner more selective. For example, scan_net
will now only carve pcap files that are IPv4 or IPv6.

* scan_net now properly reports ethernet addresses and TCP connections into the files ether.txt and tcp.txt.

TCP memory structure scanning is now disabled by default; it can be explicitly enabled with -Scarve_tcp=true.

* scan_elf applies more validation to ELF headers and, as a result, many of the false positives have been eliminated. This was expecially a problem for truncated headers.

* scan_winpe applies more validation to section and DLL names and, as a result, many of hte false positives have been eliminated. This was especially a problem for truncated PE headers.

* Strings of hex digits separated by colons are no longer mistaking reported as Ethernet MAC addresses.

* compression bomb logic improved. bulk_extractor will no longer decompress the same compressed block twice. As a result, histograms are now based on distinct occurances, rather than multiple occurances within the same compressed object.

==========
UNDERREPORTING FIXES

* Default max_depth for recursion changed from 5 to 7. (We found some deep content on our test drives).

* scan_pdf has been improved and now does a better job extracting text. For example, the following feature is now found when scanning the ubnist1.gen3 disk:

1114252989-ZIP-2353-PDF-2413 202-395-1181 Karen Evans at 202-395-1181.

==========
BUG FIXES

* scan_zip now properly reports the date and time of zip components. It also reports in ISO8601 format, which makes parsing easier.

==========
USABILITY IMPROVEMENT

* The -R option will no longer process a directory that contains a file with a .E01, .000 or a .001 extension. Unfortunately, some users were using -R to process a directory filled with image files. The program will still process such files if they are in a subdirectory.

==========
INTERNAL IMPROVEMENTS

* There has been additional refactoring of the plugin system. Many functions and variables that were previously global or static are now in the be13:plugin:: namespace.

==========
PERFORMANCE STATISTICS

This section tracks how performance of bulk_extractor has changed over time. We update it with each new release

NOTE: We recommend compiling Version 1.4 with -O3 under GCC. Please use version GCC 4.7 or above. One a Mac you may wish to use clang.

Use these configure flags to compile with a different optimization:

--without-opt Drop all -O C flags
--without-o3 Do not force O3 optimization; use default level

Disk image: /corp/nps/drives/nps-2009-ubnist1/ubnist1.gen3.E01
/corp/nps/drives/nps-2009-ubnist1/ubnist1.gen3.E02

Media size: 1.9 GiB (2106589184 bytes)
MD5: 49a775d8b109a469d9dd01dc92e0db9c

Hardware: MacBook Pro 2 Ghz Intel Core i7, 8GB 1333 Mhz DDR3
512GB SSD (Simson's Laptop "Mucha"),

Current and Historic Times with no tuning [1]:

MacOS 10.8.0; LLVM build 2336.11.00; -O3

version 1.4: 144 seconds (14.59 MBytes/sec) (-O3;)
version 1.3: 185 seconds (11.34 MBytes/sec) (-O3;)
version 1.2.0: 141 seconds (14.9 MBytes/sec) (-O3;)
version 1.1.3: 171 seconds (12.3 MBytes/sec) (-O3; AES disabled)
version 1.0.7: 256 seconds (8.22 MBytes/sec) (-O3; AES disabled;

MacOS 10.7.8; LLVM build 2336.1.00; No optimization

version 1.2.0: 350 seconds (5.72 MBytes/sec)
version 1.1.3: 468 seconds (4.28 MBytes/sec)

Windows 7, same hardware ("boot camp"):

version 1.4: TBD
version 1.3: 198 seconds (10.6 MBytes/sec) (-O3; AES enabled; 32bit)
version 1.3: 186 seconds (11.33 MBytes/sec) (-O3; AES enabled; 64bit)
version 1.2.0: 207.4 seconds (9.69 MBytes/sec, [2])

Notes:
1 - Times reported are the fastest of three consecutive runs
2 - bulk_extractor 1.2.0 scan_exiv was disabled under Windows

Current list of bulk_extractor scanners:

scan_accts - Looks for phone numbers, credit card numbers, etc
scan_aes - Detects in-memory AES keys from their key schedules
scan_ascii85 - TBD
scan_base16 - decodes hexadecimal test
scan_base64 - decodes BASE64 text
scan_bulk - TBD
scan_elf - Detects and decodes ELF headers
scan_exif -
scan_exiv2 - Decodes EXIF headers in JPEGs using libexiv2 (for regression testing)
scan_email -
scan_exif - Decodes EXIF headers in JPEGs using built-in decoder.
scan_find - keyword searching

scan_gps - Detects XML from Garmin GPS devices
scan_gzip - Detects and decompresses GZIP files and gzip stream
scan_hashid
scan_hiber - Detects and decompresses Windows hibernation fragments
scan_json - Detects JavaScript Object Notation files
scan_kml - Detects KML files
scan_lightgrep
scan_net - IP packet scanning and carving
scan_pdf - Extracts text from some kinds of PDF files
scan_rar
scan_vcard - Carvees VCARD files
scan_windirs - TBD
scan_winpe -
scan_winprefetch - Extracts fields from Windows prefetch files and file fragments.
scan_wordlist - Builds word list for password cracking
scan_xor
scan_zip - Detects and decompresses ZIP files and zlib streams

Current list of bulk_extractor feature files:

aes_keys.txt - AES encryption keys
alerts.txt - Features found on alert list (redlist)
ccn.txt - credit card numbers
ccn_track2.txt - Track 2 information
domain.txt - All extracted domain names and IP addresses
email.txt - extracted email addresses
ether.txt - extracted ethernet addresses. Currently
overcollecting due to a failure to consider
local context.
exif.txt - All exif fields from JPEGs; extracted as XML.
find.txt - Hits on find command.
hex.txt - Notable hexdecimal constants
gps.txt - Extracted GPS coordinates from Garmin XML and
GPS-enabled JPEG files
ip.txt - Extracted IP addresses from scan_net
cksum-bad indicates checksum test failed;
those are less likely to actually be IP
addresses.
json.txt - Extracted and validated JavaScript Object
Notation fragments.
kml.txt - Extracted KML files
rar.txt -
report.xml - The DFXML file that explains what happened.
rfc822.txt - All extracted RFC822 headers
tcp.txt - Summaries of all extracted UDP and TCP packets.
telephone.txt- Extracted phone numbers
url.txt - Extracted URLs
url_facebook-id - extracted Facebook IDs
url_microsoft-live - extracted Microsoft Live IDs
url_searches - extracted search terms
url_services - extracted services from URLs
winprefetch.txt - Windows prefetch files and fragments,
recoded as XML for easy processing.
wordlist.txt - All the words
zip.txt - Information about all ZIP files and zip
components.

Current list of carving directories:
unrar/
jpeg/

unzip/

==========
Planned Feature List for 1.5:

* scanner to remove all XML tags (extracts text from .docx files)

* improvements to identify_filename so it can run on a report in place.

* atomic map object

* replace as many and sets maps as possible with unordered equivallent.

* Find more false positives with CCN validator by scanning through XOR data

* identify_filenames should present the % of disk that has allocated files.

* Lnk File decoder (http://msdn.microsoft.com/en-us/library/dd871305.aspx)
for automatically detecting and reporting Window shortcut files.

* xz, 7zip, and LZMA/LZMA2 decompression

* BZIP2 decompression

* CAB decompression

* Outlook Compressible Encryption (OCE) decryption

* Scanning for the start of bitlocker protected volumes.

Architecture improvements:

* introduce a concept of an atomic set and map to avoid the use of
cppmutexes in the callers. This could be used for the
feature_recorder_map and the seen_set in the feature_recorder_set.h

We are also considering the following scanners (and need help!):

* NTFS decompression

* Better handling of MIME encoding

* SQLite database identification (first survey - how fragmented are they?)

* Python Bridge.

Validation Improvements:

* Process more data with -e xor and look for CCN hits. Most will be false positives

We have tabled the following ideas, for the following reasons:

* Support on distributed computing arrays. (May be less important
given the low cost of 64-core machines)

* Support for checkpointing using BLCR. The project was abandoned.

__._._._____

___,_._,___