Query Overview

CB v4.2.5.150311.1434

March 11, 2015

Contents

Highlights

- full boolean support with `and`, `or` and `-`
- nested boolean support with parenthesis. e.g., `(foo or bar) baz`
- multiple terms are `AND`'ed if not otherwise specified
- wildcard searches with `*`, e.g., `process_name:*.exe`
- force phrase searches with double-quotes: `"foo\bar"`
- terms without fields are expanded to search all default fields (see below)
- terms can be limited to a single field with `field:term`-style syntax, e.g., `process_name:svchost.exe`
- searches are (generally) case-insensitive
- terms are whitespace delimited, so use double-quotes or escape whitespaces when required. e.g., `filemod:"c:\program files\"` or `filemod:c:\program\ files\"`

Query Syntax Details

Terms, phrases and operators

A *term* is a single keyword (without whitespace) that will be searched in the CB process or binary data store:

```
foo
```

Terms can be combined by logical operators and be nested to form more complex queries:

- and, AND, or whitespace: boolean AND operator: `foo bar, foo and bar`
- or, OR: boolean OR operator: `foo or bar`
- -: boolean NOT operator: `-foo`
- nesting using parenthesis: `(foo or bar) baz`

Terms can be also combined to form phrases. A phrase is a set of terms separated by whitespace but enclosed in quotes. Whitespace between the terms of a phrase is not treated as logical AND operator; rather a phrase is searched as a single keyword: `"foo bar"`

Phrases can be combined and nested with other phrases and terms using logical operators: `"foo bar" or baz`

Restrictions on terms

Whitespace Because whitespace is the default delimiter, a query with whitespace would be parsed as multiple terms, e.g.

Input: `c:\program files\windows`

Becomes: `c:\program` and `files\windows`

A phrase query can be submitted to avoid automatic parsing, e.g.

Input: `"c:\program files\windows"`

Becomes: `"c:\program files\windows"`

Alternatively whitespaces can be escaped using backslash (\) as the escape character, e.g.

Input: `c:\program\ files\windows`

Output `c:\program files\windows`

Parenthesis Because parenthesis is used as a delimiter for nested queries, a query with parenthesis would be parsed as a nested query, and if a proper nesting can not be found, a syntax error would be returned, e.g

Input: `c:\program files (x86)\windows`

Becomes: `c:\program` and `files` and `x86` and `\windows`

A phrase query can be submitted to avoid automatic nesting, e.g.

Input: `"c:\program files (x86)\windows"`

Becomes: `"c:\program files (x86)\windows"`

Alternatively whitespaces can be escaped using backslash (\) as the escape character, e.g.

Input: `c:\program\ files\ \(x86\)\windows`

Becomes: `c:\program files (x86)\windows`

Negative sign Because negative sign is used as logical NOT operator, queries that begin with a negative sign would be negated in the submitted query, e.g.

Input: `-system.exe`

Becomes: not `system.exe`

A phrase query can be submitted to avoid automatic negation, e.g.

Input: `"-system.exe"`

Becomes: `"-system.exe"`

Double Quotes Because double quotes are used as a delimiter for phrase queries, a query with double quotes has to be escaped using backslash (\) e.g.

```
cmdline:"\"c:\program files \(x86\)\google\update\googleupdate.exe\" /svc"
```

Fields

Below is a complete list of fields searchable by Carbon Black. Fields are valid in either the process search or binary search; some are valid in both. Any binary-related field used in the process search searches the executable backing the process.

If no field is specified for a term, the search will be executed on all default fields. Default fields are indicated by (def) the search type.

Field	process search	binary search	field type	description
md5	x (def)	-	md5	MD5 of the process, the parent, a child process, a loaded module or written file.
domain	x (def)	-	domain	Network connection to this domain.
ipaddr	x	-	ipaddr	Network connection to/from this IP address.
modload	x (def)	-	path	Path of module loaded into this process.
filemod	x (def)	-	path	Path of a file modified by this process.
regmod	x (def)	-	path	Path of a registry key modified by this process.
path	x (def)	-	path	Full path to the executable backing this process.
process_name	x (def)	-	keyword	Filename of the executable backing this process.
parent_name	x (def)	-	keyword	Filename of the parent process executable.
childproc_name	x (def)	-	keyword	Filename of the child process executables.
cmdline	x (def)	-	cmdline	Full command line for this process.
hostname	x (def)	-	keyword	Hostname of the computer the process executed on.
host_type	x (def)	-	keyword	Type of the computer: workstation, server, or domain controller.
group	x (def)	-	keyword	Sensor group this sensor was assigned to, at the time of process execution.
username	x (def)		keyword	User context the process executed with.
process_md5	x (def)	-	md5	MD5 of the executable backing this process.
parent_md5	x (def)	-	md5	MD5 of the executable backing the parent process.
filewrite_md5	x (def)	-	md5	MD5 of a file written by this process.
childproc_md5	x (def)	-	md5	MD5 of the executable backing the created child processes.
modload_count	x	-	count	Total count of module loads by this process.
filemod_count	x	-	count	Total count of file mods by this process.
regmod_count	x	-	count	Total count of registry mods by this process.
netconn_count	x	-	count	Total count of network connections by this process.
childproc_count	x	-	count	Total count of child processes created by this process.

start	x	-	datetime	Start time of this process in computer's local time.
last_update	x	-	datetime	Last activity in this process in computer's local time.
last_server_update	x	-	datetime	Last activity in this process in server's local time.
process_id	x	-	long	The internal Carbon Black process guid for the process.
parent_id	x	-	long	The internal Carbon Black process guid for the parent process.
sensor_id	x	-	long	The internal Carbon Black sensor guid of the computer this process executed on.
watchlist_<id>	x	x	datetime	The time this process/binary has matched the watchlist query with <id>.
os_type	x	x	keyword	Type of the operating system: windows, osx or linux.
md5	-	x (def)	md5	The binary's md5.
orig_mod_len	x	x	count	Size in bytes of binary at time of collection.
copied_mod_len	x	x	count	Number of bytes collected.
is_executable_image	x	x	bool	True if the binary is an EXE (versus DLL or SYS)
is_64bit	x	x	bool	True if architecture is x64.
observed_filename	x	x (def)	path	Full path of the binary at the time of collection.
digsig_publisher	x	x (def)	text	If digitally signed, the publisher.
digsig_issuer	x	x (def)	text	If digitally signed, the issuer.
digsig_subject	x	x (def)	text	If digitally signed, the subject.
digsig_prog_name	x	x (def)	text	If digitally signed, the program name.
digsig_result	x	x (def)	sign	If digitally signed, the result.
digsig_sign_time	x	x	datetime	If digitally signed, the time of signing.
product_version	x	x (def)	text	Product version string from FILEVERSIONINFO
file_version	x	x (def)	text	File version version string from FILEVERSIONINFO
product_name	x	x (def)	text	Product name string from FILEVERSIONINFO
company_name	x	x (def)	text	Company name string from FILEVERSIONINFO
internal_name	x	x (def)	text	Internal name string from FILEVERSIONINFO
original_filename	x	x (def)	text	Original name string from FILEVERSIONINFO
file_desc	x	x (def)	text	File description string from FILEVERSIONINFO
product_desc	x	x (def)	text	Product description string from FILEVERSIONINFO
comments	-	x (def)	text	Comment string from FILEVERSIONINFO
legal_copyright	x	x (def)	text	Legal copyright string from FILEVERSIONINFO
legal_trademark	x	x (def)	text	Legal trademark string from FILEVERSIONINFO

private_build	x	x (def)	text	Private build string from FILEVERSIONINFO
special_build	x	x (def)	text	Special build string from FILEVERSIONINFO
server_added_timestamp	-	x	datetime	The time this binary was first seen by the server.

Datatypes

domain

Domains are split into labels. Separator characters (the `.`) are maintained, to enable position dependent searches. A search with leading or trailing `.`'s is position-dependent. Searches with inner `.`'s are phrase searches. Searches without `.`'s will match any domain with that label anywhere in the domain name.

search	`foo.com`	`foo.com.au`
`domain:com`	**match**	**match**
`domain:.com`	**match**	no match
`domain:.com.`	no match	**match**
`domain:com.`	no match	no match
`domain:foo.`	**match**	**match**
`domain:foo.com`	**match**	no match

ipaddr

IP Addresses are searched with CIDR notation: `(ip)/(netmask)` if the netmask is omitted, it is presumed to be 32. e.g., `ipaddr:192.168.0.0/16` or `ipaddr:10.0.1.1`

text

`text` fields are tokenized on whitespace and punctuation. Searches are case-insensitive.

The string `Microsoft® Visual Studio® 2010` from the product_name field will be split into the terms `microsoft`, `visual`, `studio` and `2010`. Searches for any one of these strings will match on the binary. Phrase queries for any two consecutive terms will also match on the binary: `product_name: "visual studio"`

count

An integer value. If it exists, values from 0 to `MAXINT`. Supports two types of search syntaxes:

- `X`: Matches all fields with precisely X. e.g., `modload_count:34` for processes with exactly 34 modloads
- `[X TO Y]`: Matches all fields with counts >= X and <= Y. e.g., `modload_count:[1 TO 10]` for processes with 1 to 10 modloads

In both cases, either X or Y may be replaced the wildcard `*`. e.g., `netconn_count:*` for any process where the netconn_count field exists. `netconn_count:[10 TO *]` for any process with more than 10 network connections.

datetime

datetime fields have five types of search syntaxes:

- YYYY-MM-DD : match all entries on this day. e.g, start:2013-12-01 for all processes started on Dec 1, 2013
- YYYY-MM-DDThh:mm:dd : match all entries within the next 24 hours from this date and time. e.g, start:2013-12-01T22:15:00 for all processes started between Dec 1, 2013 at 22:15:00 to Dec 2, 2013 at 22:14:59
- [YYYY-MM-DD TO YYYY-MM-DD]: match all entries between. e.g, start:[2013-12-01 TO 2013-12-31] for all processes started in Dec 2013
- [YYYY-MM-DDThh:mm:ss TO YYYY-MM-DDThh:mm:ss]: match all entries between. e.g, start:[2013-12-01T22:15:00 TO 2013-12-01:23:14:59] for all processes started in Dec 1, 2013 within the given time frame
- -Xh: relative time calculations. Matches all entries with a time between NOW-10h and NOW. Units supported are h: hours, m:minutes, s:seconds (as observed at the host). e.g, start:-24h for all processes started in the last 24 hours.

Like with counts, YYYYMMDD may be replaced the wildcard *. e.g., start:[2013-01-01 TO *] for any process started after 1 Jan 2013.

keyword

These are text fields with no tokenization. The term searched for must exactly match the value in the field. e.g., process_name:svchost.exe. Queries containing wildcards can be submitted with keyword queries. e.g., process_name:*.exe

md5

These are keyword fields with an md5 sum. The term searched for must exactly match the value in the field. e.g., process_md5:6d7c8a951af6ad6835c029b3cb88d333

path

path fields are special text fields. They are tokenized according to path hierarchy. e.g., path:c:\windows.

For a given path, four most common subpaths are tokenized: filename, path root, full path and parent path. e.g., c:\windows\system32\boot\winload.exe is tokenized as: winload.exe, c:, c:\windows\system\boot\winload.exe and c:\windows\system\boot. For queries involving path segments that are not tokenized, wildcard queries can be submitted. e.g., path:*system*, for any path that has system as sub-path in it.

bool

Only two possible values, the string true or false. Searches are case-insensitive.

sign

One of the eight possible values: Signed, Unsigned, Bad Signature, Invalid Signature, Expired, Invalid Chain, Untrusted Root, Explicit Distrust. Values with whitespace must be enclosed in quotes. e.g., digsig_result:Signed or digsig_result:"Invalid Chain"

cmdline

Command line strings that contain parenthesis or double quotes must be escaped using a backslash. If the string also contains whitespaces either it must be enclosed in double quotes or whitespaces must be escaped as well. e.g., `cmdline:"\"c:\program files \(x86\)\google\update\googleupdate.exe\" /svc"` or `cmdline:\"c:\program\ files\ \(x86\)\google\update\googleupdate.exe\"\ /svc`

Example Searches

Process Example Searches

Example Query Strings	Result
`domain:www.carbonblack.com`	Returns all processes with network connections to/from domains matching the given FQDN
`domain:.com`	Returns all processes with network connections to/from domains matching `*.com`
`domain:.com.`	Returns all processes with network connections to/from domains matching the form `*.com.*`
`domain:www.`	Returns all processes with network connections to/from domains matching the form `www.*`
`domain:microsoft`	Returns all processes with network connections to/from domains matching `*.microsoft OR *.microsoft.* OR microsoft.*`
`ipaddr:127.0.0.1`	Returns all processes with network connections to/from IP address `127.0.0.1`
`ipaddr:192.168.1.0/24`	Returns all processes with network connections to/from IP addresses in the network subnet `192.168.1.0/24`
`modload:kernel32.dll`	Returns all processes that loaded a module with matching path (accepts path hierarchies)
`modload:c:\windows\system32\sxs.dll`	Returns all processes that loaded a module with matching path (accepts path hierarchies)
`regmod:\registry\machine\system\controlset001\control\deviceclasses*`	Returns all processes that modified a registry entry with the matching path (accepts path hierarchies)
`path:excel.exe`	Returns all processes with the matching path (accepts path hierarchies)
`path:c:\windows\system32\notepad.exe`	Returns all processes with the matching path (accepts path hierarchies)
`cmdline:backup`	Returns all processes with matching command line arguments
`hostname:win-5ikqdnf9go1`	Returns all processes executed on host with matching hostname

group:"default group"	Returns all processes executed on hosts with matching group name (use of quotes are required when submitting two-word group names)
host_type:workstation	Returns all processes executed on hosts with matching type (use of quotes are required when submitting two-word host types)
username:system	Returns all processes executed with the matching user context
process_name:java.exe	Returns all processes with matching name
parent_name:explorer.exe	Returns all processes executed by a parent process with matching name
childproc_name:cmd.exe	Returns all processes that executed a child process with matching name
md5:5a18f00ab9330ac7539675f3f326cf11	Returns all processes, modified files, or loaded modules with matching MD5
process_md5:5a18f00ab9330ac7539675f3f326cf11	Returns all processes with matching MD5
parent_md5:5a18f00ab9330ac7539675f3f326cf11	Returns all processes that have a parent process with given MD5
filewrite_md5:5a18f00ab9330ac7539675f3f326cf11	Returns all processes that modified a file or module with matching MD5
childproc_md5:5a18f00ab9330ac7539675f3f326cf11	Returns all processes that executed a child process with matching MD5
<type>_count:*	Returns all processes that have xxx_count field > 0, where type is one of modload, filemod, regmod, netconn, childproc
<type>_count:10	Returns all processes that have xxx_count field = 10, where type is one of modload, filemod, regmod, netconn, childproc
<type>_count:[10 TO 20]	Returns all processes that have xxx_count field >= 10 and <= 20, where type is one of modload, filemod, regmod, netconn, childproc
<type>_count:[10 TO *]	Returns all processes that have xxx_count field >= 10, where type is one of modload, filemod, regmod, netconn, childproc
<type>_count:[* TO 10]	Returns all processes that have xxx_count field < 10, where type is one of modload, filemod, regmod, netconn, childproc
start:2011-12-31	Returns all processes with a start date of 2011-12-31 (as observed at the host)
start:[* TO 2011-12-31]	Returns all processes with a start date earlier than or equal to 2011-12-31 (as observed at the host)
start:[* TO 2011-12-31T22:15:00]	Returns all processes with a start date earlier than or equal to 2011-12-31 at 22:15:00 (as observed at the host)

`start:[2011-12-31 TO *]`	Returns all processes with a start date later than or equal to 2011-12-31 (as observed at the host)
`start:[2011-12-31T09:45:00 TO *]`	Returns all processes with a start date later than or equal to 2011-12-31 at 09:45:00 (as observed at the host)
`start:*`	Returns processes with any start date (as observed at the host)
`start:[* TO *]`	Returns processes with any start date (as observed at the host)
`start:-10h`	Returns all processes with a start time between NOW-10h and NOW. Units supported are, h: hours, m:minutes, s:seconds (as observed at the host)
`last_update:2011-12-31`	Returns all processes last updated on date 2011-12-31 (as observed at the host)
`last_update:[* TO 2011-12-31]`	Returns all processes last updated on a date earlier than or equal to 2011-12-31 (as observed at the host)
`last_update:[* TO 2011-12-31T22:15:00]`	Returns all processes last updated on a date earlier than or equal to 2011-12-31 at 22:15:00 (as observed at the host)
`last_update:[2011-12-31 TO *]`	Returns all processes last updated on a date later than or equal to 2011-12-31 (as observed at the host)
`last_server_update:[2011-12-31T09:45:00 TO *]`	Returns all processes last updated on a date later than or equal to 2011-12-31 at 09:45:00 (as observed at the server)
`last_server_update:*`	Returns processes with any update date (as observed at the server)
`last_server_update:[* TO *]`	Returns processes with any update date (as observed at the server)
`last_server_update:-10h`	Returns all processes last updated between NOW-10h and NOW. Units supported are, h: hours, m:minutes, s:seconds (as observed at the server)
`process_id:<guid>`	Returns the process with given process id, where `<guid>` is a signed 64-bit integer
`parent_id:<guid>`	Returns the process with the given parent process id, where `<guid>` is a signed 64-bit integer
`sensor_id:<guid>`	Returns processes executed on host with given sensor id, where `guid` is a unsigned 64-bit integer

Binary Example Searches

Example Query Strings	Result

`md5:5a18f00ab9330ac7539675f3f326cf11`	Returns all binaries with matching MD5
`digsig_publisher:Oracle`	Returns all binaries with a digital signature publisher field with matching name
`digsig_issuer:VeriSign`	Returns all binaries with a digital signature issuer field with matching name
`digsig_subject:Oracle`	Returns all binaries with a digital signature subject field with matching name
`digsig_prog_name:Java`	Returns all binaries with a digital signature program name field with matching name
`digsig_result:"<status>"`	Returns all binaries with a digital signature status of `<status>`
`digsig_sign_time:2011-12-31`	Returns all binaries with a digital signature date of 2011-12-31
`digsig_sign_time:[* TO 2011-12-31]`	Returns all binaries with a digital signature date earlier than or equal to 2011-12-31
`digsig_sign_time:[2011-12-31 TO *]`	Returns all binaries with a digital signature date later than or equal to 2011-12-31
`digsig_sign_time:*`	Returns binaries with any digital signature date
`digsig_sign_time:[* TO *]`	Returns binaries with any digital signature date
`digsig_sign_time:-10h`	Returns all binaries with a start time between NOW-10h and NOW. Units supported are, h: hours, m:minutes, s:seconds.
`<type>_version:7.0.170.2`	Returns all binaries with matching version, where `<type>` is product or file
`product_name:Java`	Returns all binaries with matching product name
`company_name:Oracle`	Returns all binaries with matching company name
`internal_name:java`	Returns all binaries with matching internal name
`original_filename:mtxoci.dll`	Returns all binaries with matching filename
`observed_filename:c:\windows\system32\mtxoci.dll`	Returns all binaries that have been observed to run or loaded with the given path
`<type>_mod_len:[* TO 10]`	Returns all binaries that have `<type>_mod_len` (module length in bytes) field < 4096, where type is orig or copied
`<type>_desc:"database support"`	Returns all binaries that have `<type>_desc` field with matching text, where type is file or product
`legal_<type>:Microsoft`	Returns all binaries with matching `legal_<type>` field text, where type is trademark or copyright

`<type>_build:"Public version"`	Returns all binaries with matching `<type>_build` field text, where type is special or private
`is_executable_image:True or False`	Boolean search (case insensitive) returning all binaries that are executable/not executable
`is_64bit:True or False`	Boolean search (case insensitive) returning all binaries that are 64-bit/not 64-bit

Alliance Search Fields

Any document matching an Alliance feed is tagged with an `alliance_score_<feed>` field, where the value is a score from 1 to 100. `<feed>` is the “short name” of the Alliance feed, such as “nvd”, “isight”, or “virustotal.” For any Alliance feed, you can click the “View Hits” button to discover the feed’s short name.

Example Query Strings	Result
`alliance_score_<feed>:*`	Returns all binaries that have `<feed>` score > 0
`alliance_score_<feed>:10`	Returns all binaries that have `<feed>` score = 10
`alliance_score_<feed>:[10 TO 20]`	Returns all binaries that have `<feed>` score >= 10 and <= 20
`alliance_score_<feed>:[10 TO *]`	Returns all binaries that have `<feed>` score >= 10
`alliance_score_<feed>:[* TO 10]`	Returns all binaries that have `<feed>` score < 10