TA-DB$_CyberTechDocumentation$

Release 1.0.0

Jun 19, 2018

Table of Contents

Latest documentation is posted on http://ta-db-networks.readthedocs.io

CHAPTER 1

Overview

1.1 About the TA

The DB Networks DBN-6300 uses syslog to provide event reporting to a central Security Information and Event Management (SIEM) system and to report general system health information. Syslog output is encoded in the Common Event Format (CEF), which allows easy integration into a number of common security information and event management (SIEM) andlog-analysis tools. DB Networks can provide sample integration with popular tools. This manual describes the DBN-6300 syslog messages.

App Author: - Brandon Kirklen – Email - Splunk Answers - Github

1.2 Splunk/DBN Version Compatibility

Splunk Version	App Version	DBN Version
Splunk 6.5.2	1.0.0	2.2.14

1.3 Install From Github

This TA will soon be avalible on SplunkBase, check back for more. Currently you can clone this github repo into your *$SPLUNK_HOME* folder and then restarting Splunk Enterprise.

Clone::

```
git clone https://github.com/DBNetworks/TA-DB_Networks.git TA-DB_Networks
```

CHAPTER 2

Installation and Configuration

2.1 Installation Steps

2.1.1 Step 1: Install the App

Install the DB Networks TA by downloading the latest release from

2.2 Configure TA-DB_Networks

CHAPTER 3

User Guide

3.1 DBN TA Source Types

Splits incoming feed into:

1. `system_counters`: This source type is used for various system counter information including
    - `cnt`: An external dump of the internal counter's page, lists stats for incoming feed and engine processing
    - `sys`: Contains system level information including free memory, cache, and system uptime
    - `slowsys`: A more complete set of system level information including airflow readings, disk usage, and wear indicators
    - `dbfwsys`: Information specific to the DBFW process running
2. `sqli_events`: SQL injection events will be associated with this sourcetype. This includes two subevent types
    - `distinct_event`: description of the first sql statement which is deemed a potential sql injection attack
    - `repeat_event`: events which match an injection on a statement already alerted on
3. `discovery_events`: These alerts are triggered in response to new events within the flows being monitored but without rising to the level of an attack.
    - `mds_new_user`: A new user is seen for the first time
    - `mds_new_service`: a new service is seen for the first time
    - `mds_new_host`: a new host is seen for the first time
    - `mds_new_listener`: a new listener is seen for the first time
    - `tally_new_ipseity`: a new context is seen linking client and servicer in dimensions (tally board, user, service, client, server)
4. `health_events`: contains events mainly involving engineering metrics
    - `heart_beat`: used to monitor system up status on a more frequent basis than `dbfwsys`
    - `engine_start`: used to monitor for engine restarts

- `archive`: Indicates status of overnight system archive tool
- `dbfw_gc`: Indicates a system restart due to overload of data
- `dbdu`: postgres database disk usage

5. `insider_threat_events`: events related to table level analysis preformed with the insider threat module
6. `audit`: events exported by native device auditing
7. `upgrade`: raw dump of upgrade messages for external viewing
8. `internal`: catch for bad output of internal messages, trashed

CHAPTER 4

DBN 6300 Syslog Messages

4.1 DBN 6300 Message Overview

Syslog messages forwarded from the DBN-6300 are formatted to meet the CEF header specification. As of 2.2.14, dbn output will begin phasing out CEF label support in favor of custom fields directly in the syslog message.

Syslog Message Format:

CEF Header Field	DBN-6300 Data
Version	0
Device Vendor	DB Networks
Device Product	ADF
Device Version	Current system version
Signature ID	Numeric ID
Name	String name associated with ID
Severity	Value from 0-10, system specified
cs1Label	system identifier
cs1	System serial number
system_identifier	System serial number
Message	Varies by event

4.2 Signature ID and Name Values

Note: The default size of the rsyslog is 8K. Logs that exceed this size are truncated automaticaly. If you expect syslog messages greater than 8K, increase the default message size to avoid truncation.

4.3 DBN 6300 Syslog Message Detail

4.3.1 Engine Restart Message

The restart message the startup of the DBN-6300. This message indicates that the DBN-6300 has completed its power up sequence after an initial power-up, restart/reset, or fatal error. If this message is detected and no intentional restart was initiated, contact customer service to investigate the cause.

A typical message resembles the following:

```
<133>2017-03-14T18:57:33.449245-05:00 dbfw adf: CEF:0|DB Networks|ADF|2.2.13|3|engine_
↪start|
5|cs1Label=system identifier cs1=00:00:00:00:00:00
```

The message is identified by Signature ID= `3` and name= `engine_start`.

4.3.2 Event Report Messages

Event report messages are generated as soon as an event is detected. There are two types of event report messages:

- `distinct_event` messages pertain to new unique SQL statements that are detected as possible threates. Distinct events have a Signature ID= `0` and name= `distinct_event`
- `repeat_event` messages represent repeated executions of previously detected SQL statements. Repeat events have a Signature ID= `1` and name= `repeat_event`

Both messages contain the same information, but are distinguished by the labels above appearing in the name field of the CEF prefix.

A typical `distinct_event` resembles the following. A `repeat_event` has the same structure, but the `cnt` field is greater than 1.

```
<133>2017-03-14T19:27:32.883848-05:00 dbfw adf: CEF:0|DB Networks|ADF|2.2.
↪13|0|distinct_event|
5|cs1Label=system identifier cs1=00:00:00:00:00:00 externalId=2737 cnt=1␣
↪rt=1489537652883
start=1336602182934 destinationServiceName=master cn1Label=statement identifier␣
↪cn1=2736
cat=structural dst=10.10.10.77 dpt=1305 src=10.10.10.186 spt=3585 cs2Label=score␣
↪cs2=0.500
cs3Label=confidence cs3=likely act=exec_dispatch target_sql_id=1099
```

The first part of the message contains the elements of the standard CEF format. The table below describes the event-specific fields.

Field	Description
externalId	Unique event id used to look up the event in the DBN Logs
cnt	Number of occurances of events with given statement identifer
rt	Transmit time of the event
start	epoch time of event (milliseconds)
destinationServiceName	Name of the database associated with the attack
cn1Label	Statement Identifier
cn1	Unique statement id
cat	type of event (structural or parametric)
dst	Destination IP
dpt	Destination Port
src	Source IP
spt	Source Port
cs2Label	Score
cs2	Numerical confidence score (normalized between 0-1)
cs3Label	Confidence
cs3	String confidence description (certain, overwhelming, likely, suspicious, possible)
act	Type of action involved (Maps to protocol RPC)
target_sql_id	Integer value represented on the system by the target SQL ID

4.3.3 System Health Messages

Health syslog messages are sent every 10 minutes (at minute mod 10 boundaries). These messages are distinguished from event messages by the keywords `cnt`, `sys`, `slowsys`, and `dbfwsys` in the CEF Name field. These messages contain system information useful to DB Networks' Customer Support personnel.

Example `cnt` message:

```
<133>2017-03-14T19:27:30.140860-05:00 dbfw adf: CEF:0|Engineering|ADF|Dev␣
↪Build|11|cnt|0|
cs1Label=system identifier cs1=unknown rt=1489537650140 xtime_T01=03/14/17 18:57:32
xtime_T02=03/14/17 19:27:30 xtime_T03=1 xtime_T04=0:29:58 xtime_T05=05/09/12 17:22:36
xcap_X13=0 xcap_X01=1928135 xcap_X02=0 xcap_X33=0 xcap_X14=0.00% xcap_X03=0 xcap_X12=0
xcap_X26=1291 xcap_X27=13847 xcap_X28=13478 xcap_X04=100.00% xcap_X15=0 xcap_X11=3307
xcap_X21=0.00% xpro_X08=369 xpro_X34=0 xpro_X17=57 xpro_X22=0 xpro_X23=0.00% xpro_
↪X24=0.00%
xpro_X25=0 xpro_X16=312 xpro_X06=0 xpro_X10=0 xpro_X05=0.00% xpro_X09=0.00% xpro_
↪X18=863932
xpro_X19=99.95% xpro_X20=0.06% xpro_X07=0 xeng_X29=1111 xeng_X30=381678 xeng_X31=758
xeng_X32=62 zpro_Z04=0 zpro_Z05=0.00% zpro_Z06=0.00% zpro_Z08=0.00% zpro_Z09=0.00%
zpro_Z07=0.00% zpro_Z01=0 zpro_Z02=0 zpro_Z03=0 zpro_Z10=0.00% zpro_Z11=0 zpro_Z12=0
zpro_Z13=0 zpro_Z14=0 zpro_Z15=0.01% zpro_Z16=349 zpro_Z17=5 zpro_Z18=0 zpro_Z19=0
zpro_Z22=-1960 zpro_Z23=0 zpro_Z24=359 zpro_Z25=4 zpro_Z26=6 zpro_Z27=0 zpro_Z28=0
zpro_Z29=0 zpro_Z30=0 ts=1489537650138
```

As with event messages, the first part of the messages contains the elements defined in the CEF format. Through most of the information in the various health log messages is useful only to DB Networks' support, there are a few fields which can be mapped useful external concepts.

Useful Event Message Counters:

- `xcap_X13` : Total number of packets recieved on the capture port. If this number is not increasing as expected for a given installation, the capture port might not be capturing traffic.

- `xcap_X15` : Total number of packets dropped by the engine. If this number increase rapidly, it might indicate that the span/tap port is configured to send a lot of non-sql traffic. This affects system preformance and should be corrected either by changing the span/tap port configuration or adjusting the network filters on the DBN-6300 to filter out unwanted traffic before it reaches the engine.

The following messages are also sent every 10 minutes. These messages can be useful to DB Networks customer support and development personnel if an issue arises.

`sys:`

```
<133>2017-03-14T19:27:32.144918-05:00 dbfw adf: CEF:0|Engineering|ADF|Dev␣
↪Build|12|sys|0|
cs1Label=system identifier cs1=unknown rt=1489537652144 os_uptime=19946 os_loadavg_0=1
os_loadavg_1=1 os_loadavg_2=0 os_freemem=940785664 os_totalmem=8339775488 sys_
↪user=346203
sys_nice=153 sys_system=69859 sys_idle=7503488 sys_iowait=33658 sys_irq=7909 sys_
↪softirq=7575
sys_steal=0 sys_guest=0 sys_guest_nice=0 vm_pgpgin=1636152 vm_pgpgout=14213713
vm_pswpin=0 vm_pswpout=0 vm_pgfault=126057794 meminfo_MemTotal=8144312 meminfo_
↪MemFree=918736
meminfo_MemAvailable=6537620 meminfo_Buffers=300864 meminfo_Cached=5421820 meminfo_
↪SwapCached=0
meminfo_Active=4977304 meminfo_Inactive=1817392 meminfo_Active(anon)=874512 meminfo_
↪Inactive(anon)=292504
meminfo_Active(file)=4102792 meminfo_Inactive(file)=1524888 meminfo_Unevictable=0
meminfo_Mlocked=0 meminfo_SwapTotal=976892 meminfo_SwapFree=976892 meminfo_Dirty=1788
meminfo_Writeback=0 meminfo_AnonPages=1072004 meminfo_Mapped=481036 meminfo_
↪Shmem=95012
meminfo_Slab=347176 meminfo_SReclaimable=296968 meminfo_SUnreclaim=50208
meminfo_KernelStack=6960 meminfo_PageTables=25816 meminfo_NFS_Unstable=0
meminfo_Bounce=0 meminfo_WritebackTmp=0 meminfo_CommitLimit=5049048 meminfo_Committed_
↪AS=3431092
meminfo_VmallocTotal=34359738367 meminfo_VmallocUsed=0 meminfo_VmallocChunk=0
meminfo_HardwareCorrupted=0 meminfo_AnonHugePages=0 meminfo_ShmemHugePages=0
meminfo_ShmemPmdMapped=0 meminfo_CmaTotal=0 meminfo_CmaFree=0 meminfo_HugePages_
↪Total=0
meminfo_HugePages_Free=0 meminfo_HugePages_Rsvd=0 meminfo_HugePages_Surp=0
meminfo_Hugepagesize=2048 meminfo_DirectMap4k=124736 meminfo_DirectMap2M=6154240
meminfo_DirectMap1G=2097152 memsum_usedGb=1 memsum_freeGb=6 disk_sda_readOps=70343
disk_sda_readSectors=3270248 disk_sda_writeOps=509080 disk_sda_writeSectors=28427427
```

`slowsys:`

```
<133>2017-03-14T19:27:38.146333-05:00 dbfw adf: CEF:0|Engineering|ADF|Dev␣
↪Build|13|slowsys|0|
cs1Label=system identifier cs1=unknown rt=1489537658145 disk_root_total=57521228
disk_root_avail=41946336 disk_boot_total=194235 disk_boot_avail=82772 disk_maint_
↪total=2818080
disk_maint_avail=1583852 vers=0
```

`dbfwsys:`

```
<133>2017-03-14T19:27:34.173796-05:00 dbfw adf: CEF:0|Engineering|ADF|Dev␣
↪Build|14|dbfwsys|0|
cs1Label=system identifier cs1=unknown rt=1489537654172 dbfw_pid=88958 dbfw_state=0
dbfw_userCpu=9344 dbfw_sysCpu=791 dbfw_numThread=21 dbfw_VmSize=940736512 dbfw_
↪VmRSS=524038144
```

4.3.4 New Discovery Messages

New discovery syslog messages are sent when the DBN-6300 identifies a new user, service, host, listener, or context linking client and server in dimensions (ipseity).

The fields associated with these various messages are:

Signature ID	Name	Description
6	`mds_new_user`	• `user_name` =<string = non-empty user name> • `default_schema` =<string = default schema for new user>
7	`mds_new_service`	• `service_name` = <string = service_name> • `service_name_type` =<string =service type (service|SID|global name)> • `dialect` =<string = database dialect (Oracle|MS Sql)>
8	`mds_new_host`	• `realm` =<string = realm name> • `addr` =<string =IPV4 address>
9	`mds_new_listener`	• `realm` = <string = realm name> • `addr` = <string = IPV4 address> • `port` = <integer = TCP/IP port>
10	`tally_new_ipseity`	• `tally_board` = <string = identifier for tally board, currently main> • [`user_name` = <string = non-empty user name>] • [`service_name` = <string = non-empty service name] • `client_realm` = <string = client realm name> • `client_addr` = <string = IPV4 addr of client> • `server_realm` = <string = server listener realm name> • `server_addr` = <string = IPV4 addr of server listener> • `server_port` = <int = TCP/IP port of server listener> • `client_ipseities` = <int = pre-existing ipseities with matching client host – zero implies this is the first> • `server_ipseities` = <int = pre-existing ipseities with matching server host> • [`server_service_ipseities` = <int = pre-existing ipseities with matching server host

Example Messages:

`mds_new_user`

```
<133>2017-03-14T19:00:22.970916-05:00 dbfw adf: CEF:0|DB Networks|ADF|Dev Build|8|mds_
↪new_user|5|
cs1Label=system identifier cs1=none rt=1489536022968 realm=default user_name=XXCC␣
↪default_schema=XXCC
```

`mds_new_service`

```
<133>2017-03-14T19:27:14.737219-05:00 dbfw adf: CEF:0|DB Networks|ADF|Dev Build|7|mds_
↪new_service|5|
cs1Label=system identifier cs1=00:00:00:00:00:00 rt=1489537634735 service_name=master
service_name_type=service dialect=Sql Server
```

`mds_new_host`

```
<133>2017-03-13T19:52:09.712603-05:00 dbfw adf: CEF:0|DB Networks|ADF|Dev Build|8|mds_
↪new_host|5|
cs1Label=system identifier cs1=00:00:00:00:00:00 rt=1489452729711 realm=default␣
↪addr=10.0.0.1
```

`mds_new_listener`

```
<133>2017-03-14T19:00:22.988379-05:00 dbfw adf: CEF:0|DB Networks|ADF|Dev Build|9|mds_
↪new_listener|5|
cs1Label=system identifier cs1=00:00:00:00:00:00 rt=1489536022980 realm=default␣
↪addr=10.0.0.1 port=1305
```

`tally_new_ipseity`

```
<133>2017-03-14T19:00:28.548773-05:00 dbfw adf: CEF:0|DB Networks|ADF|Dev␣
↪Build|10|tally_new_ipseity|5|
cs1Label=system identifier cs1=00:00:00:00:00:00 rt=1489536028542 tally_board=main␣
↪service_name=master
client_realm=default client_addr=10.0.0.1 server_realm=default server_addr=10.0.0.2
server_port=1163 client_ipseities=0 server_ipseities=1 server_service_ipseities=0
```

4.3.5 Insider Threat Event Messages

Insider threat messages are sent when the DBN-6300 sees statement executions meeting the criteria of an insider threat rule that has been configured to monitor and syslog. The purpose of these messages is alert customers to policy and stability violations in a monitored network. Insider threat rules are defined in terms of sets or patterns describing data flows. A data flow is the unique combination of a partially or fully qualified table name (for example, “master.sys.databases” specifies database, schema, and relation, but not server) mentioned in a specific network context (i.e., client IP, server IP, server Port, database service, and database user). When a statement is executed, the DBN-6300 analyzes the SQL text semantically, looks up the corresponding data flow (or flows if there are more than one qualified name in the statement), and checks whether that flow meets the criteria of an insider threat rule. If the rule’s action is configured to write to syslog when it fires, the details of the data flow and unique identifiers for several aspects of the flow and rule are conveyed in messages described below.

Example:

```
<133>2017-03-14T19:21:21.109481-05:00 dbfw adf: CEF:0|DB Networks, Inc.|ADF|Dev␣
↪Build|18|it_threat_event|5|
cs1Label=system identifier cs1=00:00:00:00:00:00 eventId=2174 tix_id=1855 tix_
↪annotation=New general rule 1489537194368
spec_id=1857 spec_type=general spec_annotation=Catchall cat_id=222 category_name=sys.
↪syslog
flow_id=1424 start=1336601400 end=1336601400 rt=1336601400 server= database=master␣
↪schema=
relation=filerepository mode=read user_id=293 user_name=sa service_name=master
dialect=Sql Server clientIP=10.0.0.1 serverIP=10.0.0.2 listener_port=1305
```

As with the other messages described above, the first part of the message contains the elements of the standard CEF format. The event-specific fields are described in the following table.

Field	Description
eventId	Unique event ID used to look up the event in the DBN event log.
tix_id	Rule system transaction ID. Rule configuration changes are transactional, so each rule belongs to one transaction only.
tix_annotation	Rule system transaction annotation.
spec_id	Unique identifier for the rule that generated the event.
spec_type	Type of rule that generated the event. This can be ‘general’, ‘stable-qname-contexts’, or ‘stable-context-qnames’ for policy, stable qualified name, or stable context rules.
spec_annotation	Optional user annotation of the rule.
cat_id	Unique identifier for the set of actions taken when the rule associated with this event fires.
category_name	Name for the set of actions taken when the rule associated with this event fires. Typically, this is ‘sys.syslog’.
flow_id	Unique identifier for the data flow ID that triggered this event.
first_seen_tid	First time the triggering data flow was observed by DBN-6300.
last_seen_tid	Last time the triggering data flow was observed by DBN-6300.
server	Server name, if specified, in the qualified name mentioned by the sql statement that triggered this event.
database	Database name, if specified, in the qualified name mentioned by the sql statement that triggered this event.
schema	Schema name, if specified, in the qualified name mentioned by the sql statement that triggered this event.
relation	Relation name (for example, table or view) in the qualified name mentioned by the sql statement that triggered this event.
mode	Mode of access (read or write) of the qualified name mentioned by the sql statement that triggered this event.
user_id	Unique identifier for the database user employed to execute that statement that triggered this event.
user_name	Database user name used to execute the statement that triggered this event.
service_name	Service name against which the statement that triggered this event was executed.
dialect	Dialect (for example, Sql Server) of the above service name.
clientIP	Client IP address employed to execute the statement that triggered this event.
serverIP	Server IP address against which the statement that triggered this event was executed.
listener_port	Port of the above server IP.

CHAPTER 5

Release Notes

5.1 Dealing With Custom Fields

5.1.1 Deprication

Current data from DBN6300 complies with the CEF standard for field values. This includes a construct of mandating a static key set neccesitating a structure for use with custom fields. This structure of `cs1Label=` followed by `cs1=` results in significant message overhead which is not needed in splunk. As such this structure is being phased out of DBN syslog messages. Current releases will continue to support exisiting pairs of key/value pairs formatted in this way but will also be adding a redundant key/value pair. For example current messages contain:

```
cs1Label=system identifier cs1=00:00:00:00:00
```

This will be replaced with:

```
cs1Label=system identifier cs1=00:00:00:00:00 system_identifier=00:00:00:00:00
```

And eventually simplified to:

```
system_identifier=00:00:00:00:00
```

5.1.2 Search Time Replacement

In the mean time, if you would like to use custom field values, you can use a search time extraction like the following:

```
* | rex mode=sed field=cs1Label "s/ /_/g" | eval {cs1Label}=cs1
```

Note: Part of the difficulty when using these custom fields is they have the potential to have spaces in the label. The `rex` part of the above search replaces those spaces before using the label as a field key. This will need to be done for any such custom field before it can be used as a field name.