
Common Event Expression

December 2007

The CEE Board

Sponsor: Sponsor

Dept. No.: DepartmentNumber

Contract No.: ContractNumber

Project No.: ProjectTask

The views, opinions and/or findings contained in this report are those of The MITRE Corporation and should not be construed as an official Government position, policy, or decision, unless designated by other documentation.

This document was prepared for authorized distribution only. It has not been approved for public release.

©2007 The MITRE Corporation. All Rights Reserved.

MITRE

Abstract

This paper defines a framework to enable collaborative efforts in the creation of an open, practical, and industry-accepted event interoperability standard for electronic systems—*Common Event Expression (CEE™)*. While acknowledging the challenges and past attempts in developing logging standards, we provide a well-defined, bounded problem description as well as a common collection of terminology with which to frame the effort. We recommend a framework to address the various components of an electronic event standard: an open format *event expression taxonomy*, *log syntax*, *log transport*, and *log recommendations*. This effort goes beyond any previous attempts to standardize the event interoperability space in that we start by defining the event language and using it as motivation for the syntax and transport, and to pose recommendations for what should be logged.

Acknowledgments

The CEE Board acts as an advisory body to help define and provide strategic direction to CEE. This document was the result of the combined effort of the founding CEE members:

MITRE	Non-MITRE
W. J. Heinbockel	A. Chuvakin, LogLogic
J. T. Judge	R. Marty, Splunk
R. M. McQuaid	

We would like to thank Anton Chuvakin and Raffy Marty, who have provided much of the motivation for, and have been very supportive of the entire Common Event Expression initiative. Their comments and criticisms are reflected throughout this entire whitepaper. Without their enthusiasm and rallying other support, CEE would be just a passing thought.

We would also like to acknowledge the members of the CEE Discussion mailing list for their feedback, insights, and for keeping us grounded amidst their confusions. Most notably, we appreciate the many comments from Stanford Whitehouse.

Table of Contents

1	Problem Description	1-1
2	Scope	2-1
3	Benefits	3-1
4	Event Interoperability Examples	4-1
4.1	Enterprise Event Management: Oil Pipeline Monitoring	4-1
4.2	Regulatory Compliance: Financial Solutions	4-4
5	Comparison to Prior Efforts	5-1
6	Proposed CEE Framework	6-1
6.1	A Common Event Taxonomy	6-2
6.2	A Common Log Transport and Syntax	6-3
6.3	Log Recommendations	6-4
7	Why CEE?	7-1
7.1	Why Should We Attempt Yet Another Log Standard? What Distinguishes CEE?	7-1
7.2	How Could CEE Benefit Me or My Company? Why Should We Support CEE?	7-1
Appendix A	CEE Roadmap	A-1
Appendix B	Terminology	B-1

List of Figures

Figure 4-1. An Example Network for Monitoring Oil Production	4-2
Figure 4-2. Correlating Correlation Engines.....	4-3
Figure 4-3. Flow of Event Information from Sensors to an Operator	4-3
Figure 6-1. CEE Architecture	6-1
Figure 6-2. Examples Mapped to CEE Components	6-2

List of Tables

Table 7-1. Motivations for Various Standard Stakeholders	7-2
--	-----

1 Problem Description

All electronic systems are capable of detecting numerous distinct **events** — observable situations or environment changes — of which only a fraction of which are ever recorded, or **logged**. The act of logging an event can occur in many ways: by writing to a text log file, via Syslog or other network logging protocol, or stored in proprietary binary files or databases to be later queried. Logs contain information about events, which can include device states, monitor readings, and a variety of other information. Logs are often identified as data logs, audit logs, alerts, alarms, audit trails, and a variety of similar terms.

In an ideal situation, the logs generated by various devices reflect a near real-time infrastructure awareness; every event should be represented. With all of the necessary event data available, IT operations should be able to aggregate, correlate, and prioritize the logs in order to detect any significant events, including anomalous or malicious behaviors, and maintain a constant state of overall awareness.

Unfortunately due to disparate logging practices and log formats of each device and vendor, the creation of such an optimal environment requires an overwhelming manual effort. In order to make use of the log data, each heterogeneous log message has to be individually interpreted and normalized into a consistent representation. Only after the logs have been processed can the consumer begin to collect regulatory-compelled data, apply custom rule sets, or perform any aggregation or correlation. Furthermore, any change in the operational environment, including the deployment of new technologies or the updating of existing ones, require operators to fully reexamine the rules and log templates of the supporting log management architecture. NIST Publication 800-92 [4] further details these issues and states that the problem stems from “inconsistent log formats,” noting that “there is no consensus in the security community as to the standard terms to be used to describe the composition of log entries and files.”

2 Scope

Common Event Expression (CEE™) is developed to address the overall problem of event representation, communication, and interpretation. Previous attempts in this area, including IDMEF and WELF, have failed in gaining adoption since they only targeted a portion of the larger issue or were tied to individual vendors. As a solution, CEE recommends industry coordination in four different areas in order to facilitate log management and analysis. The CEE components of Common Log Transport (CLT), Common Log Syntax (CLS), Common Event Expression Taxonomy (CEET), and the Common Event Log Recommendations (CELR) seek a consensus in log transportation, log syntax, event representation, and event logging recommendations for various log sources and scenarios.

Since CEE is focused on individual device-generated events, whole security incidents are outside of the scope. An incident refers to the collection of information regarding impact, time, cost, or confidence assessments; point-of-contact details; mitigation strategies; and anything related to the human-factor surrounding incidents and incident response. Often, incident reports include event logs, which may be provided in the CEE format. Therefore, incident-related efforts such as IODEF (Incident Object Description Exchange Format) are supplementary and considered to be outside the scope of CEE, even though a CEE-defined event may be incorporated into IODEF-defined incidents.

We propose that the Common Event Expression (CEE) be adopted as the industry accepted way to describe and communicate events in log files.

3 Benefits

While there are many obvious benefits of an industry event standard, including improved event correlation and response, there are other advantages that may not be as apparent. Some of these benefits will be later illustrated in oil pipeline management and the financial industry examples.

1. Easier Regulatory Compliance Efforts – CEE simplifies the task of establishing and maintaining compliance with various regulatory standards that incorporate audit or security guidelines, including PCI DSS, SOX, HIPAA, FISMA, ISO27001, ITIL, COBIT, GLBA, and others.
2. Improved Monitoring and Awareness – A log standard allows companies to more easily monitor their product lines and identify problems. Just think: one standard could be used to handle everything from recordings of financial transactions to workflow monitoring to operational troubleshooting, improving overall awareness, and allowing inefficiencies to be quickly identified and corrected.
3. Improved Security Awareness – CEE represents a large component of the “Monitor and Evaluate” portion of the COBIT structure and supports many of the management procedures present in the ITIL framework. Additionally, many organizations feed their logs to security analysis engines, such as SIMs, for data mining and correlation purposes.
4. De facto Standard for Inter-organization Communication – With every device supporting the same event log standard, there is instant interoperability potential for devices deployed across multi-national enterprises and governments.
5. Improved Code Reuse – Developers and vendors can use a single log library to support all CEE-compliant logs. The community can develop and support a single library API instead of re-architecting the log framework for each new device version; the current usage of log message dictionaries would no longer be required.
6. Vendor and Device Agnostic – Established log management infrastructures rely on the logs generated by several chosen devices, essentially locking the customer into the use of those products. The purchasing of replacements or upgrades requires a costly testing and process overhaul to even maintain an equivalent level of awareness. CEE frees customers from product dependency, enabling new devices to be quickly integrated into the current environment.

7. Reduced IT and Security Operations Costs – With a standard set of information, operations centers will not require auditors and operators to be trained in interpreting messages in product-specific languages. Fewer operators can be leveraged to manage more systems.
8. Log Message Internationalization – Standard expressions result in unambiguous interpretation. Instead of vendors needing to individually produce and maintain libraries of international log messages, CEE allows for a single application to more easily translate any CEE-compatible log record.

These benefits provide numerous advantages to a variety of people, including:

Event Producers (vendors & products) will be able to decrease cost associated with logging and reuse log libraries. Vendors could move away from encouraging developers from picking log messages on a closest-fit basis from a limited, product-specific message index. Furthermore, the generation of these log messages could be based on a single API call. Also product interoperability will increase with the others who speak with the same event expressions, resulting in satisfied customers.

Event Consumers (vendors & products) will not have to worry about handling a different event syntax and description for each new version of each product, since these discrepancies should be non-existent in products supporting this standard. There would be no longer a need to employ an event mapping team to manually interpret and handle the different events produced by different devices. Additionally, the consumers can produce better, more accurate analysis because of the availability of detailed, meaningful information.

End Users (IT and Security operations) will be able to decrease unnecessary log management overhead, and easily manage and replace unrelated systems. The log messages are more informative and understandable, permitting enhanced log analysis capabilities while ensuring all various log compliance needs are met.

4 Event Interoperability Examples

As further motivation for event standardization, we provide several non-typical examples of how CEE will benefit industry, from vendors to consumers. The first model illustrates how an event standard helps correlate and manage events across an enterprise. Afterwards, a theoretical financial corporation is used to show how can leverage CEE to assist with regulatory standards compliance. These examples were designed to be demonstrative of the increased capability that can be provided by standardizing logs, while still maintaining an obvious correlation with the IT network in a standard organization.

4.1 Enterprise Event Management: Oil Pipeline Monitoring

As in most enterprise environments, logs are monitored on a network. In this case, the network happens to be responsible for operations of an oil pipeline (Figure 4-1). Sensors installed to monitor various points along the oil pipeline report back data, such as the oil purity, throughput, pressure, and temperature. However, these various types of electronic devices used to audit the pipeline's health and status, transmit information in a variety of disparate log formats.

Meanwhile, a correlation engine can be receiving the logs from those sensors, as well as other devices including intrusion detection systems (IDS), proxy servers, anti-virus scans, and firewalls. With the supporting log infrastructure, any operator should be able to easily identify any glitch in the oil flow or the supporting network. A case where an external attacker penetrates a firewall, gains access to an internal system, and causes the temperature monitor to report anomalous data, which causes the pipeline to react by stopping the oil flow, should be detectable in near real-time, as both the pipeline and network security sensors would be logging suspicious data.

With these events using varying transports, syntaxes, structures and content, machine interpretation and correlation cannot occur. Thus, a human expert analyst is needed to make sense and to tie these records together into a coherent incident "story." Correlation engines are limited; they can only correlate across a bounded number of devices that are "known" to the engine. With each device using a different logging format and language to represent events, support by a correlation engine is a manual process of interpreting and understanding every log template.

This lack of event interoperability is becoming an intractable problem as the number of electronic systems and their generated events increase — an obstacle that can be best addressed by an accepted, industry-wide event expression standard. By adopting a unified language for expressing content of logs, any correlation engine can immediately support the logs from any device and thus automate the analysis of the incident, cutting out the most expensive, unscalable and often error-prone link from this analysis chain: skilled human analyst.

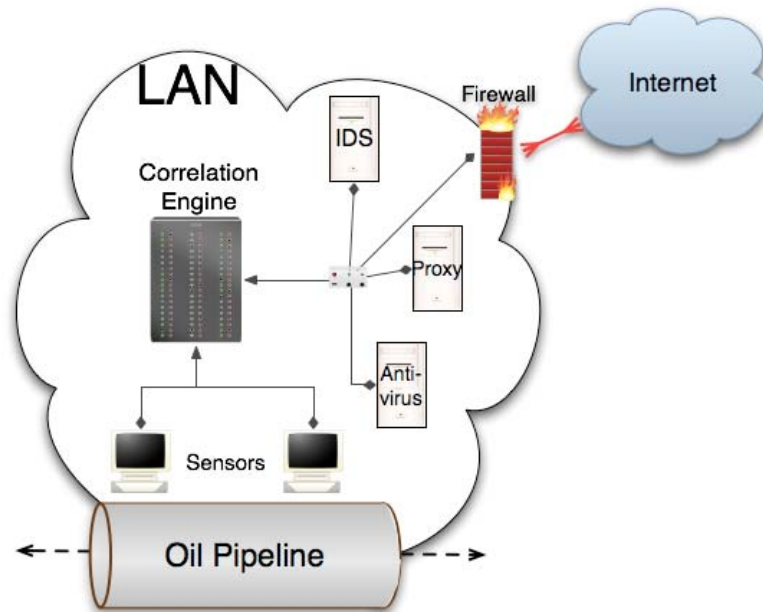


Figure 4-1. An Example Network for Monitoring Oil Production

When given the same set of logs, the resulting correlation engine report should contain similar information and use the same syntax with the same unified language. With these engines producing the same output formats with similar reports, correlation can now occur at more abstract levels resulting in summary views of the current situational awareness (Figure 4-2). Various correlation engines can be deployed across the corporation in a hierarchical structure to feed reports upward to a master correlation engine. With the oil pipeline network example, several satellite offices setup along the length of the pipeline can have their own support infrastructure while allowing headquarters to maintain an oversight over the entire operation. If there are similar or causal failures, or a coordinated attack on multiple points along the pipeline, an operator at the master correlation engine's dashboard could immediately detect and troubleshoot it.

By standardizing on the input and output log syntax and language, CEE will eliminate the continual maintenance associated with regular expression parsing and correlation engine updates.

Examining a more abstract flow of information from sensors deployed throughout an enterprise to an operator (Figure 4-3), logs can express events at different abstraction levels and are information valuable to everyone, from the operator to the CEO. Supporting such a impressive vision requires that the transport, syntax, and expression taxonomy of event log communication are standardized.

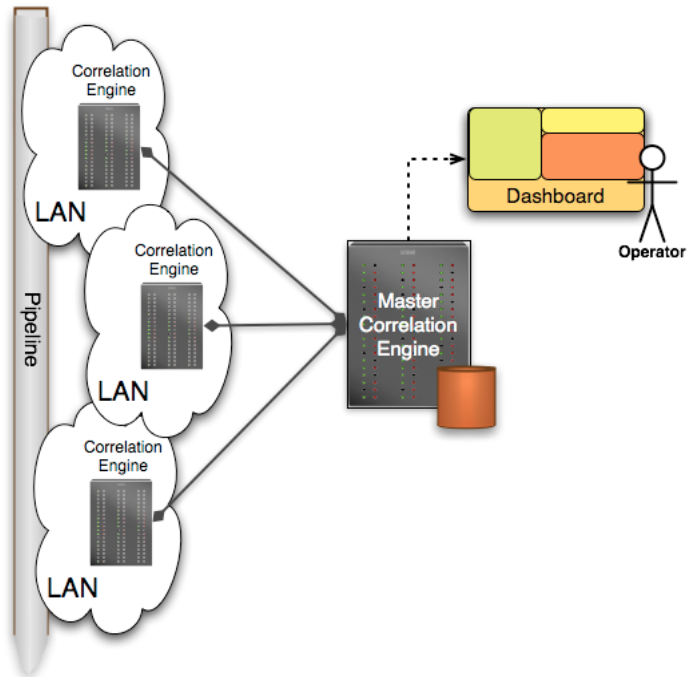


Figure 4-2. Correlating Correlation Engines

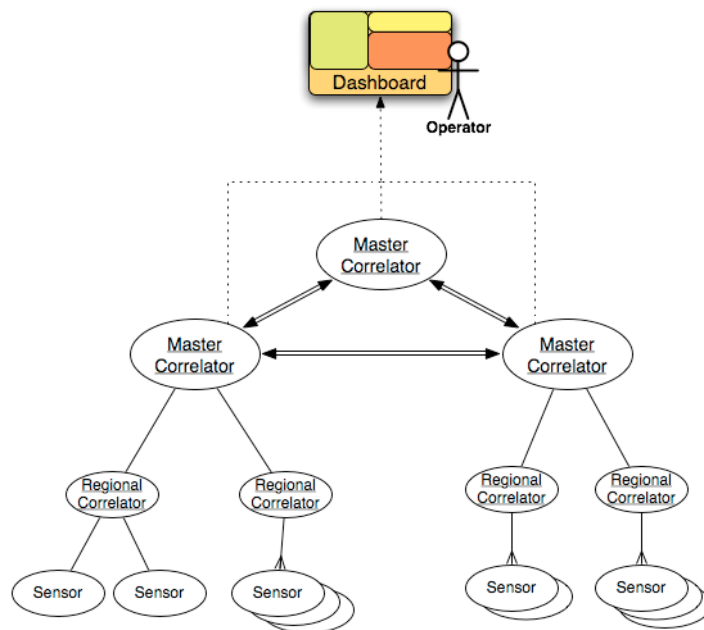


Figure 4-3. Flow of Event Information from Sensors to an Operator

4.2 Regulatory Compliance: Financial Solutions

While many industries might not have to worry about maintaining an oil pipeline, they do have to ensure compliance with various regulatory standards. PCI DSS, the Sarbanes-Oxley (SOX) and Gramm-Leach-Bliley (GLBA) Acts, HIPAA, FISMA, COBIT, ITIL, ISO27001, and the EUDPD and Basel II in Europe are acronyms with which most businesses are all too familiar. The requirements set forth carry heavy fines for non-compliance. In the end, companies are forced to spend millions of dollars to meet and maintain a satisfactory record.

In no industry is this truer than in finance. Banks, stock brokers, investment agencies, and other financially-integrated professions are required to audit each transaction while ensuring compliance with multiple regulatory standards. Even though the monetary transactions and compliance needs overlap, these aspects are handled with different systems. Do IT and system events affect the financial systems? Does the industry want to risk such an occurrence? Since companies must individually monitor and bring into compliance hundreds of internal supporting applications, log management appliances, and custom applications have become a necessity. Without any audit and event standards in place, monitoring and compliance becomes larger and more expensive burden.

To comply with current regulatory standards, each organization is required to audit various events, such as those concerning user activity or information accesses. To get this information, it must be assured that the systems are correctly logging the certain event types and that those logs are being retained. At this point, the organization is facing the problem of there not being a log standard — every device is recording the same event types in a different manner. Even though they know exactly what event types and corresponding information should be audited, there is no way to correlate each actual event log with its event type. Without a standard, it costs time and money to perform a manual review of the logs generated by each device.

Organization-wide awareness, workflow monitoring, and compliance regulations can be streamlined and more beneficial when all events are logged according to the same standard.

5 Comparison to Prior Efforts

There have been several previous attempts at developing event and log interoperability standards. For one reason or another, these efforts have not been successful in achieving industry support, some where too academic, while others were too narrowly focused. Some of the more notable efforts are highlighted below:

CIDF – The Common Intrusion Detection Framework was sponsored by DARPA and defined the Common Intrusion Specification Language (CISL) [3]. CISL was proposed in 1999 and used English-like sentence expressions and syntax trees in order to represent intrusion events. The CIDF effort was later merged in with IDMEF.

IDMEF – The Intrusion Detection Message Exchange Format is an IETF effort that followed CIDF. IDMEF [6] was designed to enable the communication of intrusion events observed by IDS devices, and consists of two entities: a syntax expressed in XML and the transport protocol (IDXP). First proposed in 2002, the most recent update occurred in 2004 and is supported by a very limited number of intrusion detection products. It also suffers from a narrow focus on intrusion event, thus unsuitable for audit logging and system troubleshooting logging.

CBE – The Common Base Event [2] model is a standard, led by IBM, that defines an XML event syntax. CBE is described as a “common language to detect, log and resolve system problems” [11] and is supported by several Tivoli products with the goal of achieving autonomic computing. After the public release of the specification and partnering with Cisco in 2003, CBE is still actively being maintained but has yet to have any noticeable industry impact, even across IBM’s own product lines.

SDEE – Security Device Event Exchange [7] was developed by the ICSA Labs and the Intrusion Detection Systems Consortium (IDSC). The SDEE XML syntax is built on the SOAP transport and appears to be only supported by Cisco. Since its introduction in 2003, there has been little done to update and support this effort.

XDAS - Distributed Audit Service [10] is a prior Open Group effort that has been reinvigorated with the help of Novell and has been renamed OpenXDAS [9]. The XDAS specification is quite large and looks to solve the log exchange problem by defining logging APIs. While the use of a common programming library with a listing of log events is a step in the right direction, there will never be a “one size fits all” programmatic solution – the standard should drive the software libraries, not vice versa. Besides the support of Novell, it is unlikely that XDAS will see their API in any major codebase.

CEF – The newest foray (September 2006) into the event syntax standards selection is the Common Event Format [1] from ArcSight. A CEF message is comprised of delimited plain text strings with optional sets of key-value pairs. It is relatively simple

to generate and parse, and is transport independent. CEF is the preferred communication method of ArcSight products, such as the Enterprise Security Manager (ESM), and is supported by several other products.

WELF – The WebTrends Enhanced Log file Format [8] is similar to CEF in that it is not bound to any specific transport and represents log data using plaintext, key-value pairs. WELF consists of four required and twenty optional syntax fields limited to expressing firewall, VPN, and other simple network-based events.

The following are included in the interest of completeness and because they are commonly and incorrectly categorized as an event standard:

IODEF – The Incident Object Description Exchange Format [5] was developed by the IETF to improve computer incident response communications and is often associated with IDMEF. Since CEE and IODEF are focused on different areas, they should be viewed as complements and not replacements for one another. IODEF is focused on the human-to-human communication of incident response, not in how the incident was discovered or the formatting of related log files. While CEE messages should be included in IODEF reports, IODEF is considered to be outside the scope of CEE.

6 Proposed CEE Framework

As a starting point for the development of CEE, we believe that the problem be best addressed as a combination of four sub-elements: log transport, log syntax, expression taxonomy, and logging recommendations, which could be worked on independently. The remainder of this document will concentrate on the three CEE components necessary to enable event exchange: the taxonomy, syntax, and transport. Once these components are defined, any CEE message can be received, parsed, and understood.

The interaction of these three elements in the conversion from events to logs (Figure 6-1) consists of an event occurring, which is represented within the taxonomy. Then, the syntax is filled with the associated details, which is transported to a receiver. Once the receiver gets a CEE message via a specific transport, all of the details are parsed from the syntax, and the event is understood with the expressed taxonomy.

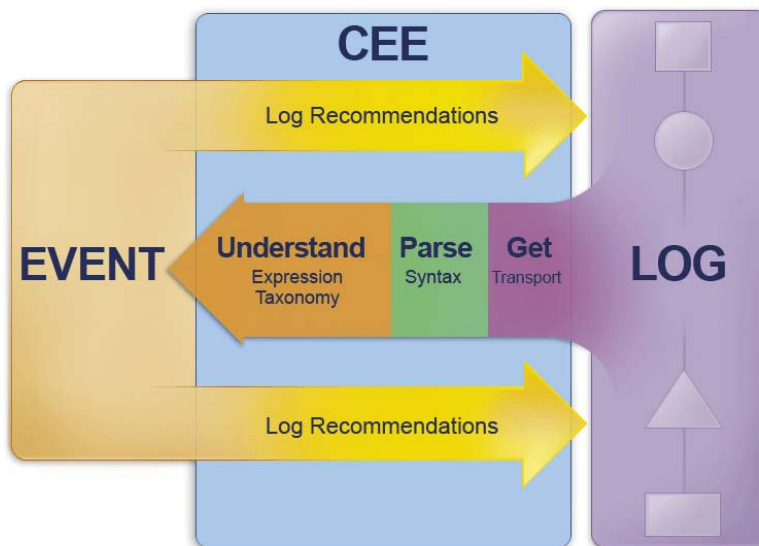


Figure 6-1. CEE Architecture

Due to the subtlety between these components, many people have difficulty in distinguishing the faint boundaries. How do practical log capabilities map into CEE? For example (Figure 7-2), if an SNMP Trap were to be translated into our CEE elements, the *transport* is the SNMP protocol version over port 162/UDP, the *syntax* is determined by the object identifier (OID) as encoded via ASN.1, and the event is identified, or *expressed*, as the Management Information Base (MIB) and is further categorized as a specific entry in common event taxonomy. Similarly, when moving logs using a proprietary SOAP-based protocol, the *transport* is HTTP, the *syntax* is defined by a specific XML schema, and the

event expression is defined in the form of a common taxonomy entry. However, preference is given towards using natural language to enhance human readability.

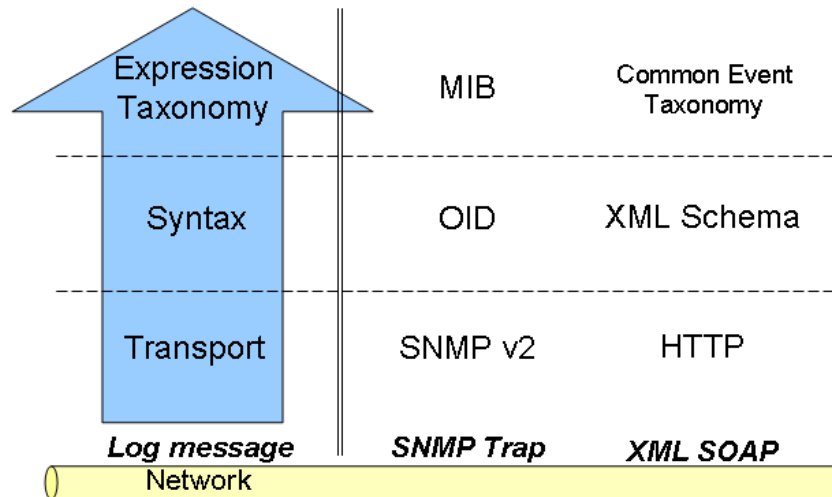


Figure 6-2. Examples Mapped to CEE Components

6.1 A Common Event Taxonomy

The Common Event Expression Taxonomy (CEET) represents the keystone of CEE. CEET is an event language — an unambiguous way of classifying logged events. If multiple systems observe the same event, it should be expected that their taxonomy description of that event is identical. A computer should be able to immediately determine whether two logs refer to the same type of event. In order to make this happen, there needs to be a collection of well-defined words that can be combined in a predictable fashion. Presumably these words would describe the type of activity, the actors involved, the outcome, and other relevant event data.

What does this mean for end-users? For example, take the simple event of the root user logging into a system. In the PAM framework, this event is expressed as “`session opened for user root by LOGIN(uid=0).`” In a typical Linux distribution it might be logged as “`ROOT LOGIN ON tty1,`” while a Snort trigger reports “`POLICY ROOT login attempt [Classification: Misc activity] [Priority: 3].`” The goal of CEE is for each of these different products to identify the event using the same terminology. Imagine how straightforward log interpretation would be if the event was always reported in the same manner, with authentication events always being represented by similar phrasing.

By defining and utilizing common taxonomy for recording events, CEE has a scalable and universal way to convey the meaning of event messages to both human and computer recipients. Event producers can be constrained to recording one event per log entry and supporting a more event consumer-focused model. By eliminating subjective information, such as perceived impact or importance, sometimes seen in current log messages, end-users and event consumers can generate a more flexible, accurate, environment-focused overview which takes into account of all possible logs from all supporting devices.

CEET may follow one of several approaches. One way is to provide a vocabulary associated with categories or “buckets” for various event characteristics. The buckets might be something like “subject,” “object,” “action type,” and so on. In each of these, the event producer selects the appropriate term. Another, similar approach would be to define a pseudo-language with subjects, objects, verbs, etc. along with a finite set of words. In this case, the producer would build a parsable grammar out of the elements, which would reflect the event details.

6.2 A Common Log Transport and Syntax

CEE makes a distinct separation between the transport and the log syntax. While the syntax is unique, it can be expressed and transmitted in a number of different ways. For example, the syntax may be expressed in XML and transported via SOAP or e-mail (SMTP). Some syntax and transport options are complementary, but others do not work as well, such as communicating XML over Syslog or SNMP. Whether the event syntax is recorded locally in a flat file (to be transported over FTP or SCP protocols in batch mode) or sent via the network on a known protocol, this choice should be left up to the event producers and consumers. Both the sender and receiver must agree as to what communication channel will be used. The Common Log Transport (CLT) is used to define the potential mediums.

The important feature of CEE standard effort is that many of the currently used log transport options may be adopted as supplemental “standard.” For example, using Syslog over port UDP 514 is used by millions of Unix-derived systems and thus can probably be “blessed” as a standard log transport mechanism.

The Common Log Syntax (CLS) defines a dictionary of syntactic identifiers to be used for communicating details regarding a logged instance of an event. Since it is not possible to create a syntax that is appropriate for every situation, the dictionary will need to define a universal set of terms along with their data types and usage (e.g., source, destination, username, domain, etc. that may be reused for previous standard efforts). By using the same data dictionary, event consumers and end-users can be assured that the expected event details are included and used consistently.

When using a syntax, there should be options, depending on the environment and objectives, as to how information is transmitted. An administrator should be able to choose the best transport, regardless of whether it is an encoded binary syntax, name-value pairs, or an

XML-based one. Here are three possibilities that address issues of speed, ease-of-use, and expressiveness.

- **Speed** – A binary log format (and corresponding syntax of fixed sized fields in a binary file) can express comprehensive information and is the fastest way to log and exchange data. When wanting to minimize size and network impact, compressed binary is the best option. However, binary syntaxes are not designed for human readability and require conversion libraries for encoding and decoding logs.
- **Ease-of-use** – Plaintext syntaxes include delimited and key-value pairs such as CSV and CEF, which humans and machines can more easily read and understand. With a fairly basic syntax, this format is very practical and would most likely have the best overall acceptance by event producers and consumers. Additionally, this type of syntax offers compatibility with a majority of transports. At the same time, this format is not as speed-efficient as a binary format above.
- **Expressiveness** – Syntaxes based on structures such as XML are comprehensive and capable of representing complex data structures, such as lists and nested object relationships. Similar to ease-of-use syntax options, an XML-based syntax would be a desirable option for some event producers and consumers. Some drawbacks include a limited choice of compatible transports as well as the overhead requires extra space for storage and transmission and possible difficulties with human understanding of such logs. Since most event data is fairly straightforward, forcing it into an expressive syntax would be unnecessarily excessive.

There have been several previous academic and commercial event standards that have been proposed, however none of them have gained any widespread usage or reached the point of being recognized as an industry standard.

6.3 Log Recommendations

The Common Event Log Recommendations (CELR) provides logging guidance of CEE initiative. With a common way of expressing events, it is possible to advocate what events products should log. While it should be expected that a firewall should log events such as blocked connection attempts, there is no logging advisements. Should firewalls log all rule change events? What about login and administration events? CELR will assure that administrators receive an entire view of all auditable events.

CELR is not only concerned with what events to log, but also addresses what information such as level of detail should be logged. This translated to “what are the various event representation elements and how should they be fulfilled?” Returning to the firewall example, there should be recommendations as to what data should be included with various firewall-related events: source, destination, NAT’ed sources, ports, protocols, and the connection result (allowed, dropped, etc.). Further considerations include how applications

should log certain events — username, source, connection method, and results for authentication; configuration changes; and a plethora of other important event information.

IDS and IPS systems need guidance concerning how to report potential attack events, such as the source, destination, what triggered the alert, and reporting with which attack that the alert is related. One important outcome of CELR regarding network sensors is whether sensors should report what they detected, what they think they observed, or both. For higher level or automated analysis, the packet details are facts and generally more useful for correlation and analysis. However, an alert guessing at a buffer overflow attack or bruteforce login attempts based upon signatures may be better for a small-scale LAN and add some input value to prioritizing. This information can then be used to as feedback to improve the CEE syntax and expression taxonomy.

7 Why CEE?

7.1 Why Should We Attempt Yet Another Log Standard? What Distinguishes CEE?

Every other effort involving event and log standardization has either too closely coupled the syntax and transport components, thus limiting usability, or has developed their standard to support a single, narrowly-defined use-case. CEE attempts to standardize the heterogeneous vocabulary so that events can be expressed in a uniform, device-independent manner. Furthermore, we realize that a single syntax is not suitable for every environment. CEE offers the vendor and operators the flexibility to choose the best option by providing several, flexible syntax and transport options, including — *which is very important!* — currently utilized transport and format options, which will be adopted by the CEE standard effort.

7.2 How Could CEE Benefit Me or My Company? Why Should We Support CEE?

In addition to the various benefits of CEE discussed in Section 3, Table 7-1 provides a breakdown as to why Software and Log Management vendors, as well as many IT endusers should support CEE. Motivational elements are broken down and provided for each of the four tenants of CEE.

Table 7-1. Motivations for Various Standard Stakeholders

	Software Vendors	IT Users	Log Management Vendors
Transport	Allow products to be compatible with existing network topologies	Identify and manage log-related traffic	Simplify log collection
Syntax	Simplify logging across all products; reduce costs of audit trail support	Simplify log analysis across all deployed products Improve system interoperability	Simplify log analysis from all deployed products Simplify searching
Taxonomy	Simplify logging across all products	Understand what is in the logs better and easier Support higher-level compliance requirements	Enable better cross-log-source analysis
Recommendations	Be able to provide the data your customers want and expect in logs	Know what audit, compliance, operational best practices are and save on research	Know what to tell their users they need to do
OVERALL	Simplify and standardize logging procedures; reduce costs of audit trail support	Reduce costs for log management and compliance overhead; improved monitoring abilities	Reduced cost for log support; improved product capabilities

Bibliography

- [1] ArcSight. "Common Event Format." Available Online:
http://www.arcsight.com/solutions_cef.htm
- [2] Bridgewater, David. "Standardize messages with the Common Base Event model." IBM developerWorks, 21 Oct 2004. Available Online:
<http://www-128.ibm.com/developerworks/webservices/library/ac-cbe1/>
- [3] Feiertag, Rich et al. "A Common Intrusion Specification Language (CISL)." DARPA, 6 May 1999. Available Online:
<http://gost.isi.edu/cidf/drafts/language19990506.txt>
- [4] Kent, Karen and Murugiah Souppaya. NIST Publication SP 800-92: "Guide to Computer Security Log Management." September 2006. Available online:
<http://csrc.nist.gov/publications/nistpubs/800-92/SP800-92.pdf>
- [5] IETF. "IETF Extended Incident Handling (INCH) Working Group." Available Online: <http://www.cert.org/ietf/inch/inch.html>
- [6] IETF. "Intrusion Detection Exchange Format." Available Online:
<http://xml.coverpages.org/idmef.html>
- [7] ICSA Labs. "The Security Device Event Exchange (SDEE)." Available Online:
<http://www.icsalabs.com/icsa/main.php?pid=jgh475fg>
- [8] Marshall. "What is the WELF log file format?" Available Online:
<http://www.marshal.com/kb/article.aspx?id=10899&cNode=5R6Q0N>
- [9] Novell. "OpenXDAS." Available Online: <http://openxdas.sourceforge.net/>
- [10] Open Group, The. "Distributed Audit Services." Available Online:
<http://www.opengroup.org/pubs/catalog/p441.htm>
- [11] XML Cover Pages. "IBM and Cisco Collaborate on Autonomic Computing Solutions." OASIS, 10 Oct 2003. Available Online:
<http://xml.coverpages.org/CiscoIBM-CBE.html>

Appendix A CEE Roadmap

In order to better support the development, coordination, and support of the CEE effort, we have developed a phased plan to track progress for each one of the CEE components:

	Phase I	Phase II	Phase III
Taxonomy	Analyze “what’s usually in the logs” and agree on a taxonomy approach (tree, tags, hierarchy, multiple, etc)	Publish a taxonomy and talk to software vendors for adoption	Increase adoption of taxonomy across various logs; have vendors map all new log messages to a taxonomy
Syntax	List a few current log formats and outline their preferred uses (e.g., binary for high-performance, XML for descriptiveness and system consumption, etc.)	“Bless” a few and maybe modify them to better fit the project goals (having adoptions of changed ones in mind)	Update the Phase II results as times change and new log source are used
Transport	List all current log transport methods	“Bless” a few of the above as standard	Work on some NG uniform log transport mechanism
Recommendations	Summarize current logging recommendations from industry and compliance guidance	Categorized logging recommendations for various scenarios and “bless” them as standard or CEE-compliance	Update as necessary

Appendix B Terminology

In the computer industry, many terms are used very loosely and therefore may have a multitude of different connotations. To avoid any ambiguities, we provide definitions for the following terms, as used within this paper:

Aggregation is the identification and combination of two or more similar log entries.

Aggregation is used to identify and remove duplicate log entries or to merge the details from log entries regarding the same event instance.

Correlation is the association of two or more log entries of unique events. Correlation can be used to group events into a series, often by time sequence or causality.

Correlation Engine is any automated piece of software capable of correlating logs (events and incidents) from multiple sources.

Events are observable situations or modifications within an environment that occurs over a time interval. An event may be a state change or reporting of an activity by a single component within a system, or may be an interaction between multiple systems. Events may occur at differing levels of abstraction and at multiple places along the log management path. As such, an event can describe an original (base) event, aggregated event, or correlated event.

Event Consumers are log management devices and analysis engines that process, store, or otherwise use logs.

Event Producers are the information systems that observe an event. This observation may be made autonomously (“an application reporting a login failure”), by an involved party in an interaction (“I received a message from another system”), or by an observational third-party (“I observed system A sending a message to system B”) such as a network sniffer or IDS.

Incident is a computer intrusion or other occurrence, usually reported by a network operations security center (NOSC), computer emergency/incident response team (CERT/CIRT), or similar. Incidents include point-of-contact, impacts, assessment, or mitigation information in addition to the standard event details. Unlike log entries, which are recorded by a machine, incident details are typically recorded by humans.

Log is the collection of one or more log entries typically written to a local log file or sent across the network to a server via Syslog, SNMP, or a custom protocol. A log may also be referred to as an audit log or audit trail.

Log entry is a single record involving details from one or more events and incidents. A log entry is sometimes referred to as an event log, event record, alert, alarm, log message, log record, or audit record. For the sake of CEE, “log entry” is synonymous with “log.”

Taxonomy is a representation of all individual components and their relationships within a finite group. The most common taxonomy is the hierarchical tree used to classify organisms, with the links representing common biological features. Within operating systems, an example would be the organization of the directories within a filesystem according to parent-child (i.e., directory-subdirectory) relationships.