Common Event Expression

June 2008

The CEE Board

Sponsor:	NATO C3A	**Project No.:**	14088004-AA
Dept. No.:	G026	**Approval No.:**	08-0953

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™). MITRE and industry, in collaboration with the NATO Consultation, Command, and Control Agency (NC3A), offer the CEE effort to standardize IT event logs. While acknowledging the challenges in developing logging standards and the contributions of past attempts, we describe a well-defined, bounded problem and provide 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 begin by defining the event language and using it as motivation for the syntax and transport, and then recommend what should be logged.

Acknowledgments

The CEE Board acts as an advisory body to help define and provide strategic direction to CEE. This document results from 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 provided much of the motivation for the CEE effort, and have been very supportive. Their comments and criticisms are reflected throughout this whitepaper. Without their enthusiasm and their rallying of other support, CEE would be merely a passing thought.

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

Table of Contents

List of Figures

List of Tables

1 Problem Description

All electronic systems can detect numerous distinct **events**—observable situations or environment changes—of which only a fraction are ever recorded, or **logged**. The actual logging of an event can occur in many ways: by writing to a text log file, by transmitting the data via Syslog or other network logging protocol, or by storing the data in proprietary binary files or databases for later query.

Logs contain information about events, which can include device states, monitor readings, and a variety of other information. Logs are often further characterized 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 would reflect a near-real-time infrastructure awareness; they should represent every event that affected a particular device. With all of the necessary event data available, information technology (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 system awareness.

Unfortunately, due to the disparate logging practices and log formats used by different devices and vendors, creating such an optimal environment requires an overwhelming manual effort. To make the log data usable, each heterogeneous log message must be individually interpreted and normalized into a consistent representation. Only after the logs have been processed can the consumer begin to collect required regulatory 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, requires operators to reexamine all 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 being developed to address the overall problem of event representation, communication, and interpretation. We propose that industry adopt CEE as the accepted way to describe and communicate events in log files.

Several academic and commercial event standards have previously been proposed in this area, including Intrusion Detection Message Exchange Format (IDMEF) and WebTrends Enhanced Log File (WELF). However, none of them has gained widespread usage or reached the point of being recognized as an industry standard because 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 to facilitate log management and analysis. The CEE components of Common Log Transport, Common Log Syntax, Common Event Expression Taxonomy, and the Common Event Log Recommendations provide a framework to achieve consensus in log transportation, log syntax, event representation, and event logging recommendations for various log sources and scenarios.

We note that CEE focuses on individual device-generated events, not on whole security incidents. The term “incident” refers to the collection of information regarding impact, time, cost, or confidence assessments; point-of-contact details; mitigation strategies; and any information related to the human factors surrounding incidents and incident response. Incident-related efforts such as IODEF (Incident Object Description Exchange Format) are therefore considered outside the scope of CEE. However, incident reports often include event logs, which may be provided in the CEE format, and a CEE-defined event may be incorporated into IODEF-defined incidents.

3 Proposed CEE Framework

As a starting point for the development of CEE, we believe that the problem can be best addressed as a combination of recommendations related to four sub-elements: log transport, log syntax, expression taxonomy, and logging. Developers could work on these sub-elements independently. This section concentrates 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 3-1) begins with an event occurring, which is represented within the taxonomy. Then the syntax is filled with the associated details, which are transported to a receiver. Once the receiver gets a CEE message via a specific transport, it parses all of the details from the syntax, and the event is understood with the expressed taxonomy.

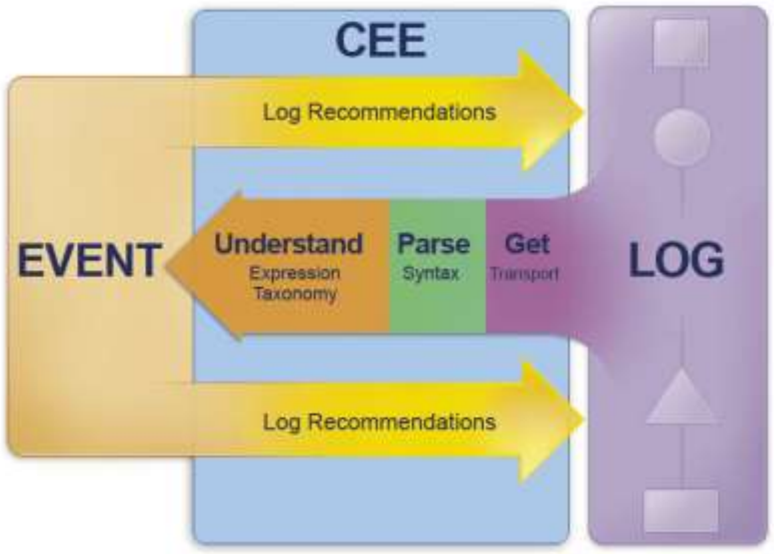

Figure 3-1. CEE Architecture

The subtle distinctions between these components make it difficult for many people to distinguish the faint boundaries between them. Figure 3-2 illustrates how practical log capabilities map into CEE. If a Simple Network Management Protocol (SNMP) Trap is translated into our CEE elements, the *transport* is the SNMP protocol version over port 162/UDP [User Datagram Protocol], 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 Simple Object Access Protocol (SOAP)-based protocol, the *transport* is HyperText Transport Protocol (HTTP), the *syntax* is defined by a specific eXtensible Markup Language (XML) schema, and the *event expression* is defined in the form of a common taxonomy entry. However, the CEE developers prefer to use natural language whenever possible to enhance human readability.

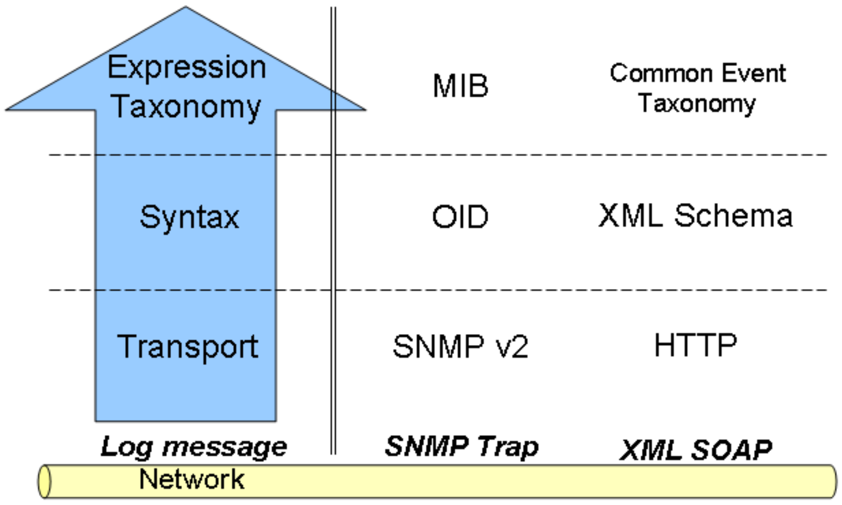

Figure 3-2. Examples Mapped to CEE Components

3.1 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, their taxonomy description of that event should be identical. Therefore, a computer should be able to determine immediately whether two logs refer to the same type of event. For this to happen, the system needs 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 CEET is for each of these different products to identify the event using the same terminology. Log interpretation would become far more straightforward if an event were always reported in the same manner, with authentication events always represented by similar phrasing. By defining and utilizing a common taxonomy for recording events, CEET offers 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 model more focused on the event consumer. By eliminating the subjective information sometimes seen in current log messages—such as perceived impact or importance—CEET allows end-users and event consumers to generate a more flexible, accurate, environment-focused overview that takes into account 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 "subject," "object," "action type," and so on. The event producer would select the appropriate term in each bucket. A 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.

3.2 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 [File Transfer Protocol] or SCP [Secure Copy] 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 on the communication channel to be used. The Common Log Transport (CLT) is used to define the potential media.

A key feature of the CEE standard effort is that many of the currently used log transport options may be adopted as supplemental "standards." For example, millions of Unix-derived systems use Syslog over port UDP 514, which 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 must 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). Using the same data dictionary assures event consumers and end-users that the expected event details are included and used consistently.

A syntax should provide options for different ways of transmitting information, depending on the environment and objectives. 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 mechanism. The following three possibilities address speed, ease of use, and expressiveness.

- **Speed** – A binary log format (and corresponding syntax of fixed-size fields in a binary file) can express comprehensive information and is the fastest way to log and exchange data. Compressed binary is the best option when the goal is to minimize size and network impact. 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 [comma-separated value] and CEF [Common Event Format], that humans and machines can more easily read and understand. With a fairly basic syntax, this format is very practical and would most likely gain the greatest 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 the binary format discussed 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, extra space for storage and transmission required by the overhead, and possible difficulties with human understanding of such logs. Since most event data is fairly straightforward, forcing it into an expressive syntax would be unnecessary.

3.3 Log Recommendations

Common Event Log Recommendations (CELR) provide logging guidance for the CEE initiative. With a common way of expressing events, it is possible to stipulate what events products should log. While it should be expected that a firewall should log events such as blocked connection attempts, there are no formal logging advisements. Should firewalls log all rule change events? Should they log login and administration events? CELR will ensure that administrators receive a comprehensive view of all auditable events.

CELR addresses not only what events to log, but also what information to capture, such as level of detail. This translates to "What are the various event representation elements and how should they be fulfilled?" Returning to the firewall example, the system should be guided by recommendations as to what data should be included with various firewall-related events: for instance, source, destination, NAT'ed [Network Address Translation] 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. Similarly, intrusion detection systems (IDS) and intrusion prevention systems would benefit from guidance concerning how to report potential attack events, such as the source, destination, what triggered the alert, and any known attack to which the alert is related.

One important outcome of CELR regarding network sensors would be a standard establishing whether sensors should report what they detected, interpretations of what they detected, or both. For higher level or automated analysis, the packet details are facts and generally more useful for correlation and analysis. However, an alert that suggests a buffer overflow attack or brute-force login attempts based upon signatures may be better for a small-scale local area network (LAN) and add some input value to prioritizing. This information can then be used as feedback to improve the CEE syntax and expression taxonomy.

4 Event Interoperability Examples

As further motivation for event standardization, we provide several atypical examples of the ways in which CEE will benefit industry, from vendors to consumers. The first model, described in Section 4.1, illustrates how an event standard helps correlate and manage events across an enterprise. In Section 4.2, we use a hypothetical financial corporation to show how a company can leverage CEE to assist with regulatory standards compliance. These examples were designed to demonstrate the increased capability that results from standardizing logs, while still maintaining an obvious correlation with the IT network in an organization.

4.1 Enterprise Event Management: Oil Pipeline Monitoring

In oil companies, as in most enterprise environments, logs are monitored on a network. In this case, the network is responsible for operations of an oil pipeline (Figure 5-1). Sensors installed to monitor various points along the oil pipeline report back data, such as the oil purity, throughput, pressure, and temperature. However, the 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 receives the logs from those sensors, as well as other devices including IDS, proxy servers, anti-virus scans, and firewalls. With the supporting log infrastructure, any operator should easily be able to identify any problems in the oil flow or the supporting network. For example, if both the pipeline and network security sensors log suspicious data, an operator should be able to detect in near-real time if an external attacker penetrates a firewall, gains access to an internal system, and causes the temperature monitor to report anomalous data, which in turn causes the pipeline to react by stopping the oil flow.

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

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 analysis of the incident, eliminating the most expensive, unscalable, and often error-prone link from this chain: the human analyst.

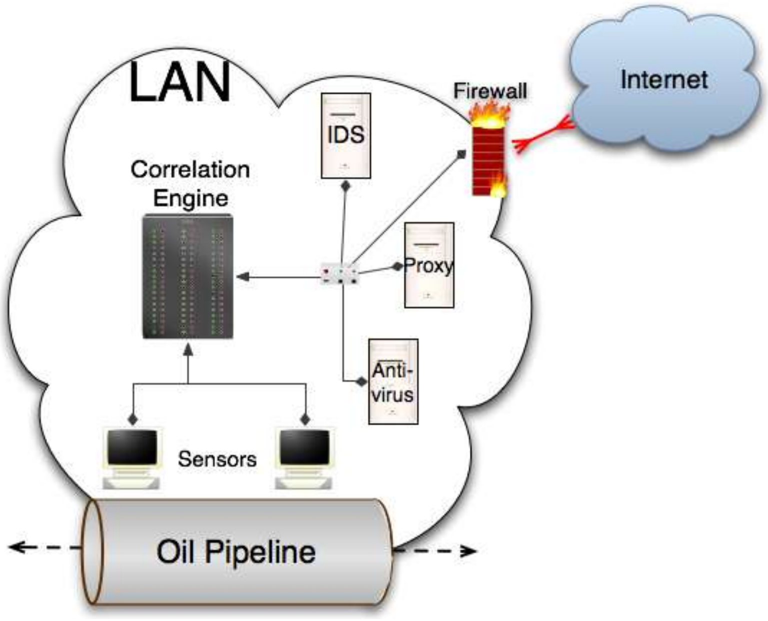

Figure 4-1. Example Network for Monitoring Oil Production

By standardizing the input and output log syntax and language, CEE would eliminate the continual maintenance associated with regular expression parsing and correlation engine updates. Given the same set of logs, the resulting correlation engine report should contain similar information and use the same syntax and the same unified language. With these engines using the same output formats for similar reports, correlation could now occur at more abstract levels, resulting in summary views of current situational awareness (Figure 4-2). Various correlation engines could be deployed across the corporation in a hierarchical structure to feed reports upward to a master correlation engine. Continuing with the oil pipeline network example, several satellite offices set up along the length of the pipeline could have their own support infrastructure, while allowing headquarters to maintain oversight of the entire operation. If similar or causal failures occur, or a coordinated attack takes place on multiple points along the pipeline, an operator at the dashboard of the master correlation engine could immediately detect and troubleshoot it.

Figure 4-3 portrays a more abstract flow of information from sensors deployed throughout an enterprise to an operator. Logs can express events at different abstraction levels and contain information valuable to everyone from the operator to the chief executive officer. Supporting such an impressive array of consumers requires standardization of the transport, syntax, and expression taxonomy of event log communication.

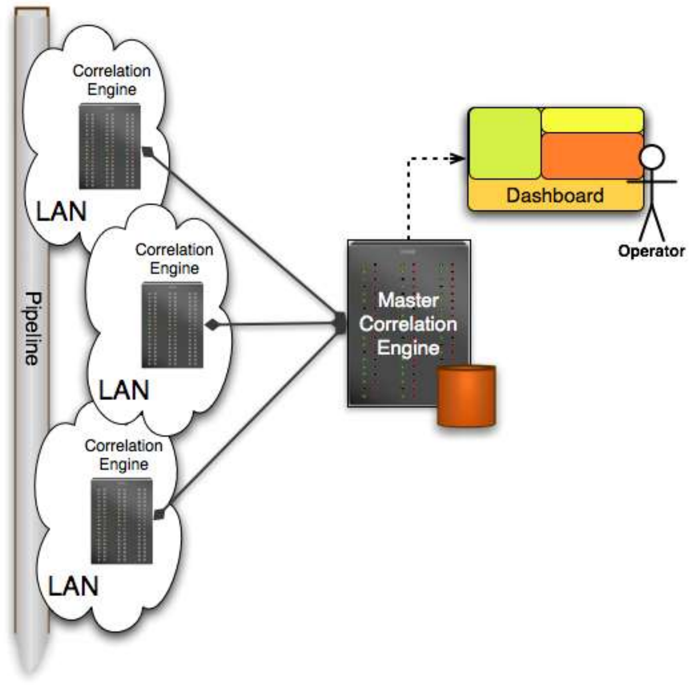

Figure 4-2. Correlating Correlation Engines

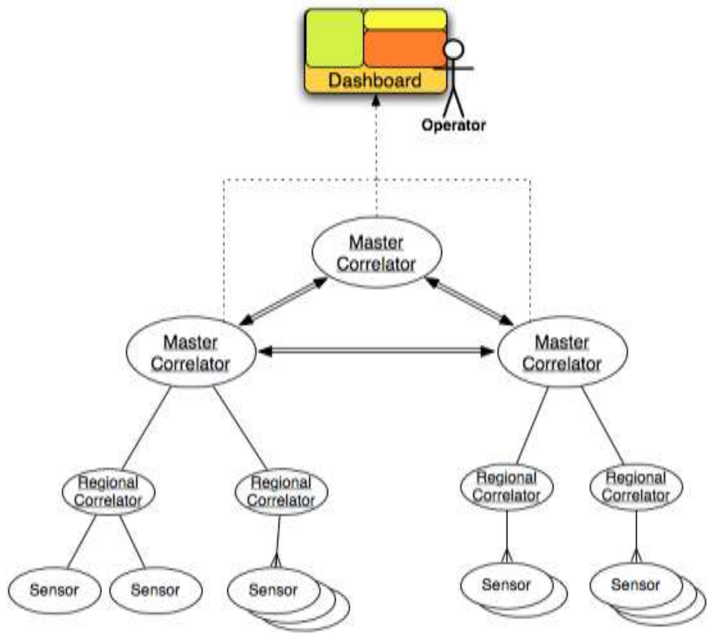

Figure 4-3. Flow of Event Information from Sensors to an Operator

4.2 Regulatory Compliance: Financial Solutions

While few industries must maintain an oil pipeline, many must comply with various regulatory standards. PCI DSS [Payment Card Industry Data Security Standard], SOX [Sarbanes-Oxley] and GLBA [Gramm-Leach-Bliley Act], HIPAA [Health Insurance Portability and Accountability Act], FISMA [Federal Information Security Management Act], COBIT [Control Objectives for Information and Related Technology], ITIL [Information Technology Infrastructure Library], ISO27001 [International Standards Organization], and the EUDPD [European Union Data Protection Directive] 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, they force companies to spend millions of dollars to meet and maintain a satisfactory record.

In no industry is this truer than in finance. Banks, stockbrokerages, 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 by different systems. IT and system events can affect financial systems, posing a risk to the industry. 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 become a 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, the industry must be certain that its systems are correctly logging the required event types and that those logs are being retained. At this point, each organization faces the problems resulting from the lack of a log standard—every device records the same event types in a different manner. Even though the company knows 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 could be streamlined and become less burdensome when all events are logged according to the same standard.

5 Comparison to Prior Efforts

There have been several previous attempts to develop event and log interoperability standards. For one reason or another, these efforts have failed to gain industry support: some were too academic, while others were too narrowly focused. Some of the more notable efforts are highlighted below.

CBE – The Common Base Event [2] model, led by IBM, is a standard that defines an XML event syntax. CBE is described as a "common language to detect, log and resolve system problems" [10] and is supported by several Tivoli products with the goal of achieving autonomic computing. Since the public release of the specification and IBM's partnering with Cisco in 2003 CBE has continued to be actively maintained, but has yet to have any noticeable industry impact, even across IBM's own product lines.

CEF – The newest foray (September 2006) into event syntax standards is the Common Event Format [1] from ArcSight. A CEF message is composed of delimited plaintext 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.

CIDF – The Common Intrusion Detection Framework, which defined the Common Intrusion Specification Language (CISL), was sponsored by the Defense Advanced Research Projects Agency (DARPA) [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 Internet Engineering Task Force (IETF) effort that followed CIDF. IDMEF [6] was designed to enable the communication of intrusion events observed by IDS devices. It consists of two entities: a syntax expressed in XML and the transport protocol (Intrusion Detection Exchange Protocol – IDXP). First proposed in 2002, with the most recent update occurring in 2004, IDMEF is supported by a very limited number of intrusion detection products. It also suffers from a narrow focus on intrusion events, and thus is unsuitable for logging audits and system troubleshooting.

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 supported only by Cisco. Since SDEE's introduction in 2003, little has been done to update and support this effort.

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, virtual private network (VPN), and other simple network-based events.

XDAS - Distributed Audit Service [9] is a prior Open Group effort that has been reinvigorated with the help of Novell. The XDAS specification is quite large and is intended to solve the log exchange problem by defining logging application programming interfaces (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. Aside from the support of Novell, it is unlikely that XDAS will see its API in any major codebase.
Update: The XDAS working group has released a 2.0 specification available on their website.

IODEF, discussed previously, is included in the interest of completeness and because it is 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 focus on different areas, they should be viewed as complements and not replacements for one another. IODEF centers on the human-to-human communication of incident response, not on how the incident was discovered or on the formatting of related log files. While CEE messages should be included in IODEF reports, IODEF falls outside the scope of CEE.

6 Why CEE?

6.1 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 its 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.

The developers realize that a single syntax is not suitable for every environment. CEE offers vendors 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 the CEE standard effort will adopt.

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

While an industry event standard confers many obvious benefits, including improved event correlation and response, such a standard offers other advantages that may be less apparent.

1. <u>Easier Regulatory Compliance Efforts</u> – 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. <u>Improved Monitoring and Awareness</u> – A log standard allows companies to monitor their product lines and identify problems more easily. One standard could be used to handle everything from recording of financial transactions to workflow monitoring to operational troubleshooting, thereby improving overall awareness and allowing inefficiencies to be quickly identified and corrected.

3. <u>Improved Security Awareness</u> – 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 SIM [Security Information Management] tools, for data mining and correlation purposes.

4. <u>De facto Standard for Inter-organization Communication</u> – Having every device support the same event log standard creates the 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 maintain even an equivalent level of awareness. CEE frees customers from product dependence, 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 no longer require auditors and operators to be trained in interpreting messages in product-specific languages. Thus, fewer operators can be leveraged to manage more systems.

8. Log Message Internationalization – Standard expressions result in unambiguous interpretation. Vendors would not need to produce and maintain libraries of international log messages individually, since CEE allows for a single application to translate any CEE-compatible log record more easily.

These benefits would apply to a variety of people and systems:

Event Producers (vendors and products) would be able to decrease cost associated with logging and to reuse log libraries. Developers would write log messages to accurately describe the event, instead of picking the best-fit one from a limited, product-specific message index. Furthermore, the generation of these log messages could be based on a single API call. Product interoperability will also increase as other systems come to use the same event expressions, resulting in satisfied customers.

Event Consumers (vendors and products) would not have to handle a different event syntax and description for each new version of each product, since none of these discrepancies should exist in products supporting the standard. Consumers would no longer need to employ an event mapping team to manually interpret and handle the different events produced by different devices. Additionally, consumers could produce better, more accurate analysis thanks to the availability of detailed, meaningful information.

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

Table 6-1 supplements the discussion by providing a breakdown as to why software and log management vendors, as well as many IT end-users, should support CEE. Motivational elements are categorized and provided for each of the four components of CEE.

Table 6-1. 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 log content better and more easily Support higher-level compliance requirements	Enable better cross-log-source analysis
Recommendations	Provide the data customers want and expect in logs	Know what audit, compliance, and operational best practices are; save on research	Know what to tell 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; improve monitoring abilities	Reduce cost for log support; improve product capabilities

Appendix A 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] Marshal. "What is the WELF log file format?" Available Online: http://www.marshal.com/kb/article.aspx?id=10899&cNode=5R6Q0N

[9] Open Group, The. "Distributed Audit Services." Available Online: http://www.opengroup.org/pubs/catalog/p441.htm

[10] 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 B 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 about 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 adoption of changed ones in mind)	Update the Phase II results as times change and new log sources are used
Transport	List all current log transport methods	“Bless” a few of the above as standard	Work on uniform log transport mechanisms
Recommendations	Summarize current logging recommendations from industry and compliance guidance	Categorize logging recommendations for various scenarios and “bless” them as standard or CEE-compliant	Update as necessary

Appendix C Terminology

In the computer industry many terms are used very loosely and therefore may have a multitude of different connotations. To avoid 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 occur 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 a system involved in an interaction (received a message from another system), or by an observational third party such as a network sniffer or IDS (observed system A sending a message to system B).

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 entity. 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 about 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. In the context 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.

Appendix D Acronym List

API	application programming interface
CBE	Common Base Event
CEE	Common Event Expression
CEET	Common Event Expression Taxonomy
CEF	Common Event Format
CELR	Common Event Log Recommendations
CIDF	Common Intrusion Detection Framework
CISL	Common Intrusion Specification Language
CLS	Common Log Syntax
CLT	Common Log Transport
COBIT	Control Objectives for Information and Related Technology
EUDPD	European Union Data Protection Directive
FISMA	Federal Information Security Management Act
GLBA	Gramm-Leach-Bliley Act
HIPAA	Health Insurance Portability and Accountability Act
HTTP	HyperText Transport Protocol
IDMEF	Intrusion Detection Message Exchange Format
IDS	intrusion detection system
IETF	Internet Engineering Task Force
IODEF	Incident Object Description Exchange Format
ISO	International Standards Organization
IT	information technology
ITIL	Information Technology Infrastructure Library
LAN	local area network
OID	object identifier
PCI DSS	Payment Card Industry Data Security Standard
SDEE	Security Device Event Exchange
SNMP	Simple Network Management Protocol
SOAP	Simple Object Access Protocol
SOX	Sarbanes-Oxley
UDP	User Datagram Protocol
WELF	WebTrends Enhanced Log File
XDAS	Distributed Audit Service
XML	eXtensible Markup Language