Etude comparative des formats d'alertes

Conference Paper · November 2015

CITATIONS
0

READS
137

4 authors, including:

Guillaume Hiet
CentraleSupélec
26 PUBLICATIONS **62** CITATIONS
SEE PROFILE

Hervé Debar
Institut Mines-Télécom
142 PUBLICATIONS **4,346** CITATIONS
SEE PROFILE

Etude comparative des formats d'alertes

Guillaume Hiet[1], Hervé Debar[2], Selim Menouar[3], and Vérène Houdebine[3]

[1] CentraleSupélec, Cesson-Sévigné, France,
guillaume.hiet@centralesupelec.fr
[2] Télécom SudParis, Evry, France,
herve.debar@telecom-sudparis.eu
[3] CS, Le Plessis Robinson, France,
prenom.nom@c-s.fr

Résumé Une des approches contribuant à la résilience des systèmes informatiques consiste à surveiller en continu leur fonctionnement afin de détecter les comportements indésirables (attaques, intrusions). L'objectif final est de ramener le système dans un état sain, éventuellement dans un mode dégradé, en exécutant des contre-mesures adéquates. Il est pour cela nécessaire de déployer différents composants qui doivent échanger de l'information sous forme d'alertes : sondes de détection, manager, base de données, interface de visualisation, etc. Dans ce contexte, la définition d'un format standard et ouvert pour les échanges d'alertes apparaît crucial. Ce format doit offrir une structuration et une richesse sémantique qui facilitent le classement et la contextualisation des alertes. Ces aspects sont essentiels pour optimiser le traitement automatiques des alertes (par exemple, la corrélation). L'utilisation d'un format standard facilite non seulement l'inter-opérabilité mais il permet également aux exploitants de capitaliser leurs efforts. Nous présentons dans cet article les résultats d'une étude comparative des formats d'alerte existants et nous proposons des pistes d'amélioration issues de ce retour d'expérience. Cette étude a été réalisée dans le cadre du projet RAPID SECEF qui s'intéresse notamment à proposer des améliorations au format IDMEF [1] pour l'adapter au contexte actuel et faciliter son adoption.

Keywords: supervision de sécurité, format d'alertes, IDMEF

1 Introduction

La supervision de sécurité est une approche de cyber-sécurité réactive qui participe à la Lutte Informatique Défensive. Elle consiste en premier lieu à détecter les attaques ou intrusions et à émettre le cas échéant des alertes (ou événements de sécurité). Une deuxième étape consiste à gérer automatiquement ces différentes alertes (processus de collecte, de stockage et de corrélation). Ces informations sont ensuite présentées aux personnes en charge de la sécurité du SI sous différentes formes (évolutions au fil de l'eau, rapports d'incident, tableaux de bord, résultats de recherches, etc.). L'objectif final est d'apporter une information la plus pertinente possible pour que les opérateurs de sécurité puissent mettre en œuvre des contre-mesures. Cette approche contribue à la résilience des

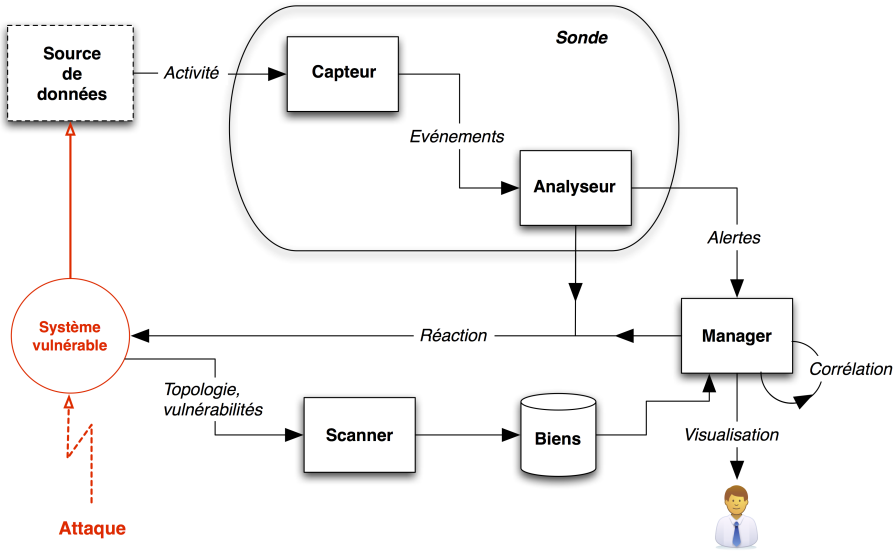

Figure 1. Architecture de supervision de sécurité

systèmes informatiques, comme le souligne la grille d'analyse de la résilience [2] proposée par le Professeur Erik Hollnagel. Celle-ci s'appuie sur quatre caractéristiques d'un système pour évaluer sa capacité de résilience. L'une d'entre-elles réside justement dans la capacité du système à surveiller les évolutions de son comportement et de son environnement.

Comme l'illustre la figure 1, la mise en œuvre d'une architecture de supervision de sécurité nécessite de déployer différents composants hétérogènes qui doivent communiquer entre eux au sein d'un *Security Operationnal Center*. Typiquement, les SOC sont constitués de différentes sondes qui doivent remonter l'information concernant les événements qu'elles ont détectés à des *managers* (*Security Information and Event Manager*). En pratique, les composants du SOC proviennent de différents éditeurs.

Dans ce contexte, la définition d'un format standard d'échange des alertes apparaît crucial. Plusieurs formats, plus ou moins spécifiques au domaine, ont été proposés par le passé mais force est de constater qu'aujourd'hui, aucun d'eux n'a été adopté massivement par les différents éditeurs. Généralement, chaque éditeur de sonde utilise un format propriétaire pour décrire les alertes émises par ses sondes. Il est donc nécessaire que le manager connaisse ces différents formats qu'il doit analyser et traduire dans un format interne, ce dernier étant lui aussi généralement propriétaire. Cette étape de normalisation est primordiale. En effet, il est nécessaire que les alertes soient exprimées dans un format commun afin de pouvoir leur appliquer facilement par la suite des traitements automatisés (corrélation, requête, etc.).

Cette situation (normalisation par le manager), n'est pas optimale. En effet, l'intégration repose alors en grande partie sur les capacités de l'éditeur du manager à normaliser de manière pertinente un grand nombre de formats d'alertes [4]. Pour les formats non reconnus nativement, l'utilisateur doit, si le produit retenu le lui permet, développer lui-même le traducteur, ce qui peut s'avérer complexe et difficile à maintenir dans le temps. Les éditeurs de managers et les utilisateurs finaux sont alors fortement dépendants des éditeurs de sondes, notamment en cas de modification du format utilisé par ces derniers.

A l'inverse, l'utilisation d'un format standard commun aux différents acteurs du domaine permettrait de s'affranchir de cette dépendance et faciliterait l'inter-opérabilité. En outre, l'utilisation par les managers d'un format standard, au lieu des multiples formats propriétaires, permettrait aux utilisateurs finaux (opérateur de sécurité, administrateur, etc.) de capitaliser leurs efforts lors du développement des traitements automatisés, tels que la corrélation. Actuellement, le développement de ces traitements nécessite d'acquérir une compétence spécifique à un produit, notamment en termes de format. Enfin, l'utilisation d'un format standard permet de développer des traitements automatiques qui soient génériques et ne dépendent pas du format propre à chaque sonde.

L'objectif du projet SECEF [5] est de promouvoir IDMEF, l'un des rares formats standards dédiés au domaine. Pour cela, le projet s'attelle à différentes tâches dont :

— l'étude comparative des différents formats plus ou moins spécifiques au domaine qui ont été proposés jusqu'à maintenant afin notamment de dégager les bonnes pratiques ;
— la proposition d'évolutions du format IDMEF, prenant notamment en compte les résultats issus de l'étape précédente, afin d'adapter le standard existant au contexte actuel ;
— la rédaction de documentation et de tutoriaux permettant de faciliter le travail des développeurs désireux d'adopter IDMEF ;
— le développement et la mise à disposition de bibliothèques permettant d'échanger et de manipuler des alertes au format IDMEF.

Nous proposons dans cet article de présenter les travaux relatifs aux deux premières tâches de ce projet : l'étude comparative des formats d'alertes et les pistes d'amélioration envisagées.

Dans un premier temps, nous rappelons en section 2 le périmètre de l'étude et la démarche retenue. Puis nous présentons la comparaison des formats en section 3 ainsi que les pistes d'amélioration en section 4. La section 5 conclut cet article.

---

4. Il s'agit d'ailleurs d'un argument commercial avancé par bon nombre d'éditeurs
5. SECEF/COSCOM est un projet financé par la DGA via le dispositif RAPID : `http://www.secef.net`

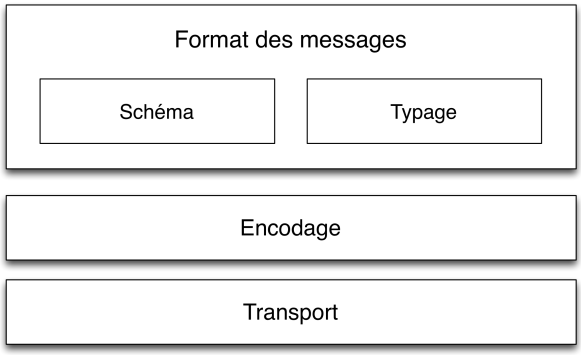

Figure 2. Format de messages

2 Périmètre de l'étude et démarche

Nous présentons par la suite les concepts relatifs à la définition d'un format de message. Nous présentons également la démarche que nous avons suivie.

2.1 Format d'alerte

Les informations remontées par les sondes vers les managers le sont sous forme de messages. Il convient donc de définir le format de ces messages. L'étude des différents formats existants fait apparaître qu'il existe en fait différents niveaux de définition pour un format donné, ce qui est illustré par la figure 2.

Classiquement, les message sont décrits par un ensemble d'attributs (ou de champs) auxquels sont associées des valeurs. Le format des messages à proprement parler comprend :

— le **schéma**, c'est-à-dire la structure des messages et la définition des différents attributs standards, ainsi que la sémantique de ces attributs ;
— le **typage**, c'est-à-dire le format des valeurs que peuvent prendre ces différents attributs.

Selon les formats, le typage peut-être plus ou moins précis (par exemple « chaîne de caractères » vs. « date au format ISO 8601 »). Le typage décrit l'ensemble des valeurs possibles pour un attribut donné. Pour certains attributs, ce typage peut prendre la forme d'une énumération ou d'un dictionnaire.

L'usage de dictionnaire apparaît comme une nécessité afin de pouvoir comparer et traiter automatiquement (notamment durant la phase de corrélation) les valeurs de certains champs issus d'alertes produites par des sondes développées par différents éditeurs. Toutefois, la définition de ces dictionnaires, c'est-à-dire d'une énumération de valeurs non ambiguës qui couvre les besoins de chacun tout en permettant de distinguer les différents cas, s'avère en pratique souvent

difficile. Il s'agit donc d'un point important dans la définition et la comparaison des formats.

Le format des messages est une spécification abstraite. L'encodage (ou format de sérialisation) détermine la manière dont les messages, c'est-à-dire les champs et les valeurs associées, vont être codés. Il est possible d'utiliser un format d'encodage ad-hoc mais il paraît préférable d'utiliser des formats génériques pré-existants (par exemple JSON ou XML). Cela permet notamment, en termes d'implémentation, de faciliter les développements (ré-utilisation de bibliothèques existantes) ainsi que de favoriser la robustesse (ré-utilisation de parseurs robustes) et l'interopérabilité.

Classiquement, on peut distinguer les encodages textuels reposant sur ASCII ou UTF (par exemple, JSON et XML) des encodages binaires (par exemple BER, CER, BSON, binary XML, etc.). Les premiers ont l'avantage d'être directement interprétables par un humain (ce qui peut être utile notamment lorsque les messages sont stockés directement dans des fichiers). Ils sont dans notre cas particulièrement adaptés car l'information à transporter est principalement de nature textuelle. Le transport d'information binaire (par exemple, capture réseau au format PCAP,) nécessite un encodage particulier, par exemple Base64 ou Uuencoding. Les encodages binaires permettent un encodage plus compact et donc d'optimiser les performances.

Le protocole de transport permet d'échanger les messages en utilisant une pile protocolaire standard (TCP/IP, étant donné le cas d'usage). Comme pour l'encodage, l'intérêt est d'utiliser des protocoles génériques existants (HTTP, SYSLOG, AMQP, etc.). Le protocole de transport et l'encodage sont en général plus ou moins couplés. Par exemple, l'utilisation de HTTP comme protocole de transport tend à favoriser l'utilisation d'encodage textuel (JSON, XML). Certains protocoles, par exemple AMQP, imposent un encodage particulier. A l'inverse, il est parfois envisageable d'utiliser différents encodages pour un même protocole (JSON ou XML sur HTTP) ou un même encodage sur différents protocoles (par exemple JSON sur HTTP ou SYSLOG).

Classiquement, le transport et l'encodage assurent un certain nombre de fonctionnalités essentielles pour le bon acheminement des messages. Ils doivent notamment fournir des fonctions de sécurité (authentification des émetteurs et des récepteurs, signature et chiffrement des messages), de haute-disponibilité (ré-émission des messages, mécanismes de redondance) et d'optimisation de la bande-passante (compression, gestion de la congestion, etc.). Ces fonctionnalités sont particulièrement importantes dans le contexte de l'échange d'alertes de sécurité. Toutefois, elles sont a priori indépendantes du format des messages à proprement parler car elles sont liées au type d'encodage et de transport utilisés.

La présente étude se focalise essentiellement sur le format de messages (c'est-à-dire le schéma et le typage). En effet, les besoins finaux (standardisation en vue de faciliter le traitement automatique des données) imposent prioritairement que le schéma et le typage soient standards. Ces derniers devraient être en grande partie agnostiques de l'encodage et du protocole de transport utilisés, le choix de ceux-ci relevant de l'implémentation.

2.2 Démarche

L'analyse de l'existant a consisté d'une part à étudier et analyser les formats que l'on souhaite promouvoir et d'autre part à les comparer à des formats similaires. L'identification de ces formats similaires a été réalisée en s'appuyant sur des études existantes, notamment celle réalisée par l'ENISA [3], l'expertise préalable des membres du consortium ainsi que les études de marché.

Nous avons comparé le format IDMEF avec 5 formats « concurrents ». Parmi ces formats, deux sont des formats propriétaires proposés par des éditeurs de SIEM :

— LEEF (IBM QRadar)
— CEF (HP ArcSight)

Les trois derniers formats sont des formats ouverts proposés par des organismes de standardisation :

— CEE (MITRE)
— XDAS/CADF (The Open Group, DMTF)
— CIM (DMTF)

Dans un premier temps, nous avons effectué une comparaison synthétique des différents formats en s'appuyant sur des critères présentés par la suite. Dans un deuxième temps, nous avons réalisé une comparaison détaillée de ces formats en établissant une table de correspondance entre les champs des différents formats. Cette deuxième étape nous a permis d'évaluer précisément la richesse sémantique de chaque format. Elle fournit également une référence pour le développement d'outils de normalisation ou de passerelles de traduction vers/depuis les formats étudiés. Enfin, elle permet de mettre en évidence les champs spécifiques à certains formats qui nous paraissent pertinents et qui constituent des pistes d'amélioration ou, à l'inverse, des champs dont la sémantique est ambiguë et/ou l'intérêt limité. Le tableau de correspondance obtenu à l'issue de cette étude est disponible sur le site du projet SECEF.

Pour l'analyse et la description synthétique des différents formats, nous avons retenus les critères suivants :

— le critère **références** permet de lister les documents de références (spécification) et donne un aperçu sur la nature et la qualité de ces documents (précision, complétude, historique, etc.) ;
— le critère **transport et encodage** permet d'indiquer quels sont les encodages et protocoles de transport utilisés (imposés par le format ou utilisés en pratique par les implémentations) ;
— le critère **pouvoir d'expressivité** permet d'identifier la nature et le type d'information que le format permet d'exprimer (sur les systèmes ciblés, la source de l'attaque, la sonde, etc.) ;
— le critère **structuration** permet de décrire brièvement la structure des messages et l'imbrication des différents attributs (format « à plat » vs. orienté objet, utilisation de l'héritage, de classes agrégées, etc.) ;
— le critère **extensibilité** permet de préciser les moyens prévus dans les standards pour étendre le format (champ additionnels, héritage, etc.).

3 Comparaison des différents formats

Dans un premier temps, nous décrivons les format étudiés en nous appuyant sur les critères énoncés ci-dessus. Dans un deuxième temps, nous présentons la synthèse des résultats de cette étude comparative.

3.1 Présentation des différents formats étudiés

CEF CEF est le format propriétaire du SIEM HP ArcSight. Il s'agit clairement d'un format orienté « événements de sécurité » en générale, c'est-à-dire non spécifique à la détection d'intrusion (il permet facilement d'exprimer l'information remontée par différents équipements de sécurité).

Références HP a publié une documentation complète mais succincte (30 pages environ) du format CEF [6]. Globalement, tous les champs sont documentés de manière claire et la documentation permet de générer une alerte CEF de manière non ambiguë. Toutefois, la sémantique exacte de certains champs est décrite de manière trop superficielle (par exemple « Application Protocol », « cat » ou « reason »).

Transport et encodage Les produits ArcSight utilisent le format Syslog comme transport/encodage. Plus précisément, un message CEF est encodé dans un champ Syslog sous la forme d'une liste d'entrées décrites par un ensemble de couples clé/valeur. Toutefois, l'utilisation d'autres encodages et transports ne semble pas présenter de difficulté technique (par exemple JSON/HTML).

Pouvoir d'expressivité Le format est assez expressif. Il permet d'exprimer les différents types d'information prévus par IDMEF (source, cible, sonde, etc.). Pour chaque type d'information, on retrouve les attributs qui paraissent essentiels. IDMEF est plus complet même si les champs non présents dans CEF semblent d'une importance plus limitée. CEF propose en outre une vingtaine d'attributs qui, à première vue, sont difficilement traduisibles en IDMEF. Certains de ces champs (par exemple, la gestion des adresses avec le NAT) constituent des pistes intéressantes pour étendre IDMEF.

Structuration Le schéma est peu structuré. Il s'agit d'une organisation « à plat » comportant 117 attributs qui n'ont pas de relations les uns par rapport aux autres. Cette absence de structure se traduit parfois par la présence de champs distincts contenant des informations similaires mais de différents types. Par exemple, les adresses IPV4, IPV6, après le NAT, etc. sont contenues dans des attributs distincts (IDMEF permet à l'inverse de spécifier un nombre illimité d'adresses et utilise un champ catégorie pour préciser le type d'adresse). Le format n'utilise pas d'attribut avec une multiplicité supérieure à un (il n'est pas possible de spécifier plusieurs sources ou plusieurs cibles dans une même alerte).

6. `https://protect724.hp.com/docs/DOC-1072`

Le format n'utilise pas de dictionnaire. Peu de champs le nécessitent mais cela serait nécessaire pour certains d'entre-eux (typiquement « outcome », « reason » ou « cat »).

Extensibilité Le format propose nativement un mécanisme d'extension assez limité : quelques champs additionnels typés mais en nombre restreint (entier, chaîne de caractère). Il est également possible d'étendre facilement le format de manière non standard en ajoutant des attributs.

LEEF LEEF est le format propriétaire du SIEM IBM QRadar. Ce « format amélioré pour les événements de journaux » (Log Event Enhanced Format) est clairement un format orienté « événements de sécurité », en particulier ceux en lien avec des problématiques de sécurité « réseau ». Le format est assez similaire à CEF dans sa structuration et dans le choix du transport et de l'encodage. En revanche, il existe des différences notables concernant le nombre et la sémantique des attributs.

Références IBM a publié une documentation complète mais très succincte (23 pages environ) du format LEEF [7]. Globalement tous les champs sont documentés mais souvent de manière trop superficielle. La sémantique de certains champs n'est pas claire, en particulier la série des champs « identXXX » (par exemple « identSrc » ou « identSecondIp »).

Transport et encodage Les produits QRadar utilisent le format Syslog comme transport/encodage de manière similaire à CEF. Toutefois, l'utilisation d'autres encodages et transports ne semble pas présenter de difficulté technique (par exemple JSON/HTML).

Pouvoir d'expressivité L'expressivité du format est limitée. La sonde est uniquement décrite en termes de noms de produits et de vendeurs (pas d'information sur les adresses, par exemple). Le format ne permet pas d'exprimer des informations sur les processus ou les fichiers (il est donc plutôt restreint aux événements « réseau »). Il ne permet donc pas d'exprimer tous les types d'informations que permet d'exprimer IDMEF. Pour les types d'informations communs (par exemple la source ou la cible de l'attaque), LEEF propose un nombre limité d'attributs permettant essentiellement de renseigner les informations relatives au réseau (par exemple, l'adresse IP).

Structuration La structuration des messages LEEF est similaire à celle de CEF. Il s'agit d'un format « à plat » comportant 50 attributs qui n'ont pas de relation les uns par rapport aux autres. Tout comme CEF, la multiplicité des attributs

7. `https://www.ibm.com/developerworks/community/wikis/form/anonymous/api/wiki/9989d3d7-02c1-444e-92be-576b33d2f2be/page/3dc63f46-4a33-4e0b-98bf-4e55b74e556b/attachment/a19b9122-5940-4c89-ba3e-4b4fc25e2328/media/QRadar_LEEF_Format_Guide.pdf`

est au plus de un et le format utilise des champs distincts pour des informations similaires de différents types (adresse IP avant/après NAT). Le format ne propose pas de dictionnaire mais peu d'attributs le nécessitent (« ressource » ou « cat »).

Extensibilité La documentation ne mentionne aucun mécanisme natif permettant d'étendre le format. Il est cependant possible d'étendre facilement le format de manière non standard en ajoutant des attributs.

CEE Common Event Expression (CEE) est un format initié par MITRE, un organisme de recherche états-unien à but non lucratif, financé principalement par des fonds publics. L'objectif (ambitieux) est de fournir un format standard pour les journaux générés par les systèmes informatiques en général. Il s'agit donc d'un domaine d'application plus large que la gestion des alertes ou événements de sécurité.

Les travaux ont été réalisés par un groupe de travail comportant des représentants de différents éditeurs et d'organismes gouvernementaux états-uniens (NIST et DoD). MITRE a joué le rôle de modérateur. Le gouvernement états-unien a depuis décidé de stopper le financement de ce projet, ce qui a conduit à l'arrêt (visiblement définitif) des travaux, d'après le site internet du projet [8].

Le groupe de travail a pris le parti de faire, dès le départ, une séparation claire entre le format des messages (CEE Event Model ou CEE Profil), l'encodage (CEE Log Syntax) et le transport (CEE Log Transport). Un profil CEE est lui-même composé d'un schéma, qui définit la structure des messages, de la spécification des différents attributs (field dictionary) et d'un ensemble de dictionnaires (event taxonomy).

Références A ce jour, les travaux réalisés n'ont conduit qu'à une version très préliminaire des spécifications. Celles-ci sont disponibles sur le site du projet CEE [9] à des fin d'archivage. La documentation est très succincte. Elle décrit essentiellement le schéma des messages sous la forme d'une page HTML, de schémas XML (XSD) et de fichiers CSV. Souvent, la sémantique des champs est ambiguë. Certains champs semblent redondants (par exemple, les champs « appname » et « app.name », « username » et « usr.name »). Il est souvent difficile de déterminer si les informations (adresse IP, nom d'utilisateur, etc.) sont relatives à la source de l'attaque, la cible ou la sonde.

Transport et encodage L'objectif de CEE était de permettre d'utiliser différents encodages et transports. A ce jour, seuls les encodages en JSON et en XML sont proposés [10]. Concernant le transport, le site dédié au format fournit un ensemble d'exigences et évoque seulement l'utilisation de Syslog avec un encodage JSON [11].

8. `https://cee.mitre.org/`
9. `https://cee.mitre.org/language/1.0-beta1/`
10. `https://cee.mitre.org/language/1.0-beta1/cls.html`
11. `https://cee.mitre.org/language/1.0-beta1/clt.html`

Pouvoir d'expressivité Le format, tel qu'il est décrit dans la version publiée sur le site de MITRE, est assez expressif. Toutefois, la sémantique exacte de beaucoup de champs est difficile à déterminer. Visiblement, le format ne permet pas de décrire la sonde de manière précise (adresse, processus, etc.) mais il est souvent difficile de savoir si les attributs sont relatifs à la source, la cible ou la sonde, chaque attributs n'étant décrits que par quelques mots. Par exemple, le champ « app » et ses attributs « app.name », « app.vend », etc. s'appliquent-t-il à la sonde ou à la cible ? Le problème se pose également pour le champ « proc » censé décrire un processus. La notion de « source » est ambiguë dans la documentation. Celle-ci évoque par exemple la notion de « event source », ce qui laisserait à penser qu'il s'agit de la sonde et non pas de la source de l'attaque. A priori, le format reprend les différents types d'informations proposés par IDMEF. Toutefois, IDMEF est plus complet et, surtout, la sémantique des champs IDMEF est moins ambiguë (par exemple, IDMEF permet clairement de préciser des informations relatives aux processus pour la sonde, la source et la cible). Il est donc difficile de traduire un événement CEE en IDMEF.

Structuration CEE est un format faiblement structuré offrant cependant une structuration intermédiaire entre les formats fortement structurés comme IDMEF et ceux reposant sur une structure totalement « à plat », comme CEF et LEEF. Il comporte 56 champs et sous-champs. Certains champs sont regroupés (utilisation de sous-champs). Par exemple, le champ « app » regroupe les sous-champs « app.name », « app.vend », etc. Le format propose quelques dictionnaires pour certains champs qui le nécessitent : « action », « domain », « object », « service » et « status ». Toutefois, certains dictionnaires mériteraient d'être complétés (par exemple « service »). En outre, certains champs associés à des dictionnaires semblent regrouper des informations très hétérogènes (par exemple « object » ou « action »).

Extensibilité En théorie, le format est extensible via l'ajout de profils additionnels (qui permettent d'ajouter des attributs et des dictionnaires). Toutefois, la documentation ne propose aucun exemple de mise en œuvre de profil additionnel.

IDMEF

Références IDMEF est décrit dans la RFC 4765. Il s'agit d'un format ouvert mais le statut de la RFC est expérimental car le groupe de travail a été dissout avant que le résultat de son travail n'ait été approuvé. L'objectif de la RFC est donc seulement de documenter le travail réalisé. Le format IDMEF est censé répondre aux exigences formalisées dans la RFC 4766. Le protocole de transport recommandé, IDXP, est décrit dans la RFC 4767 comme un « profil » BEEP.

Dans l'ensemble, le format IDMEF est bien documenté par la RFC 4765 qui spécifie la structure des messages ainsi que chacun des champs en fournissant des exemples. La sémantique de chaque champ est précisée. Le document est relativement volumineux (157 pages) ce qui traduit la complétude et la précision de la

documentation. Toutefois, sa lecture peut rebuter l'utilisateur qui souhaite créer une alerte IDMEF comportant un nombre réduit de champs (ce qui correspond à un cas d'usage assez fréquent). En outre, le format imposé par l'IETF pour les RFC (document texte ASCII) ne permet pas de faire apparaître simplement et de manière immédiate la structuration du format IDMEF (diagramme de classe complet). Enfin, si la sémantique de chaque champ est assez claire, la stratégie de peuplement d'une alerte IDMEF peut être ambiguë (ce qui est dû en partie à la grande richesse du format). Par exemple, lorsqu'une application génère des journaux qui sont par la suite analysés par une sonde HIDS, l'application en question doit-elle être considérée comme un analyseur ou la cible de l'alerte ? De telles ambiguïtés peuvent conduire à un peuplement hétérogène des alertes par des éditeurs différents, ce qui n'est pas souhaitable.

Transport et encodage La RFC 4765 n'impose pas un encodage ni un protocole de transport et, a priori, différents encodages et transports peuvent donc être utilisés. Toutefois, la RFC 4765 illustre le format en utilisant XML et fournit un schéma sous la forme d'une DTD. Par ailleurs, la RFC 4767 décrit le protocole IDXP dont l'utilisation était préconisée par le groupe de travail ayant œuvré au développement d'IDMEF. En pratique, ce protocole n'a, à notre connaissance, jamais été utilisé dans les implémentations existantes.

Pouvoir d'expressivité IDMEF est un format très riche qui permet d'exprimer différents types d'informations relatives à une alerte. Typiquement, il est possible de préciser une ou plusieurs sources d'attaques, une ou plusieurs cibles, l'attaque ou le comportement suspicieux, la sonde de détection ainsi que des informations temporelles. Il est également possible de préciser les informations relatives aux adresses réseau, aux services, processus, fichiers et utilisateurs impliqués dans la source, la cible ou la sonde.

Structuration IDMEF est un format très structuré, orienté objet, comportant 166 attributs différents répartis dans 33 classes. Beaucoup de classes sont agrégées (certains attributs sont eux-mêmes des classes). Le format a parfois recours à l'héritage mais cela est utilisé seulement pour préciser le type d'alerte (classes `CorrelationAlert`, `ToolAlert` et `OverflowAlert`) ou le type de service (classes `SNMPService` et `WebService`).

Beaucoup d'attributs sont des listes (multiplicité supérieure à un) : `Source`, `Target`, `Address`, `Reference`, etc. Cela permet de modéliser le fait qu'une attaque puisse avoir plusieurs sources ou cibles, qu'un équipement puisse avoir plusieurs interfaces (donc plusieurs adresses), etc. Ces champs sont également parfois couplés avec des champs de type « catégorie » et permettent dans ce cas de préciser différents types d'adresses ou d'identités correspondant à une même interface ou un même utilisateur. Les valeurs possibles du champ catégorie sont alors précisées dans un dictionnaire. Il s'agit d'une spécificité d'IDMEF dans le domaine des formats d'alertes. Cela souligne la richesse du format : avec seulement deux attributs, il est possible d'exprimer différents types d'adresses ou

d'identité, là où les autres formats ont plutôt tendance à dédier un attribut distinct à chaque type d'adresse ou d'identité. Cela permet également de facilement étendre le format sans augmenter le nombre d'attributs : il suffit de modifier le dictionnaire associé au champ catégorie.

De manière générale, IDMEF a souvent recours à des dictionnaires. Cela permet d'avoir des résultats facilement comparables d'une implémentation à une autre. Toutefois, ces dictionnaires ne sont pas toujours complets et nécessiteraient une mise à jour. En outre, l'usage de dictionnaires pourrait être étendu à d'autres champs (typiquement, les catégories d'attaques).

Extensibilité IDMEF propose nativement un mécanisme d'extension sous la forme d'un champ dédié (`AdditionnalData`). Il est également possible d'étendre le format de manière non standard par héritage ou en étendant les dictionnaires.

CIM Common Information Model est un standard du Distributed Management Task Force [12], un organisme états-unien de standardisation. Le DMTF est une organisation ouverte aux entreprises, organisations et personnes physiques qui développent des standards pour l'administration des systèmes informatiques distribués. Les travaux autour de CIM font l'objet de plusieurs groupes de travail au sein du DMTF. Il s'agit en effet du format central qui est utilisé par d'autres standards du DMTF (par exemple, WBEM).

CIM permet de représenter l'ensemble des composants d'un SI ainsi que les mécanismes nécessaires pour administrer et superviser ces composants. L'utilisation du format correspond donc à un périmètre beaucoup plus large que la simple supervision de sécurité. La description des alertes de sécurité représente en réalité un sous-ensemble du format, ce sous-ensemble étant, à la date de rédaction de cet article, considéré comme « expérimental ». Par la suite, la description que nous faisons se concentre sur ce sous-ensemble relatif aux alertes.

Il est à noter que CIM, dans son ensemble, est un format qui semble mature et qui a acquis une certaine notoriété pour l'administration des systèmes distribués. Il est notamment utilisé dans la solution WMI de Microsoft. Il est possible d'utiliser CIM pour décrire les composants (par exemples les sources, les cibles ou les sondes) dans d'autres formats d'alertes (XDAS semble envisager cette solution [13]).

Références CIM fait l'objet d'une spécification du DMTF (DSP). L'ensemble des documents de spécification est accessible publiquement sur le site internet du DMTF [14]. Le site référence les différentes versions du format et les documents correspondants.

12. `http://www.dmtf.org/`
13. `http://scap.nist.gov/events/2011/emapdd/presentations/EMAP%20-%20The%20Open%20Group%20Distributed%20Audit%20Services%20%28XDAS%29%20v2.pdf`
14. `http://www.dmtf.org/standards/cim`

La spécification comprend la description du méta-modèle, du langage IDL associé (MOF) ainsi que le schéma standard. Le méta-modèle est utilisé pour les évolutions du schéma ou la création d'extensions. Les différentes versions du schéma sont décrites sous forme de diagrammes UML. Ceux-ci sont fournis dans différents formats (MOF, XML, XSD, Visio et PDF). Une documentation d'API est également fournie sous forme HTML [15]. Cette documentation complète les schémas UML en décrivant les différents attributs de chaque classe.

La documentation est très complète et le format utilisé (schéma UML graphique associé à une description des différents champs) permet d'appréhender facilement et rapidement le format. Toutefois, CIM utilise souvent le concept d'héritage ce qui nécessite d'analyser les classes parentes pour identifier les champs hérités.

Le sous-ensemble de CIM relatif aux alertes de sécurité est relativement restreint et correspond au sous-schéma `CIM_Security`. Il s'agit principalement de la classe `SecurityIndication` qui étend la classe `AlertIndication`, issue du « cœur » de CIM. La classe `SecurityIndication` est héritée par la classe `IPNetworkSecurityIndication`. Cette dernière est elle-même héritée de la classe `IPPacketFilterIndication`. Ces classes sont considérées comme expérimentales.

Transport et encodage L'objectif du DMTF est de fournir un modèle abstrait standard qui puisse être implémenté dans différents langages. CIM peut donc en théorie utiliser n'importe quel format d'encodage et de transport. Le DMTF fournit un schéma XML. En outre, CIM est utilisé dans WBEM, un standard permettant la gestion à distance d'un SI distribué. Ce standard repose sur un transport HTML et un encodage XML de CIM (CIM-XML).

Pouvoir d'expressivité L'expressivité du format, en ce qui concerne les alertes de sécurité, est assez limitée. Le format ne permet pas de décrire précisément la sonde (son adresse, etc.). Aucun champ n'est prévu pour décrire les processus, les fichiers ou les utilisateurs. Cela est paradoxal car le format CIM, dans son ensemble, possède un fort pouvoir d'expressivité, permettant notamment de décrire les différents nœuds réseau et les services qu'ils hébergent. Toutefois, le schéma `CIM_Security` ne réutilise pas les classes fournies par le cœur de CIM. Seule la classe `AlertIndication` est utilisée via l'héritage.

Structuration CIM dans son ensemble est un format très structuré, suivant une approche orienté objet. Toutefois, les classes implémentant les alertes de sécurité se contentent d'hériter de la classe `AlertIndication` et n'utilisent que des champs de types « primitifs » (chaînes de caractères, entiers, etc.). Ces classes n'utilisent pas d'agrégation (aucun de leur champs ne correspond à une autre classe de CIM). On peut donc considérer que ce sous-ensemble du format est relativement peu structuré. Les classes relatives aux alertes comprennent 58 champs. Le format tend à utiliser des champs différents pour des informations

15. `http://schemas.dmtf.org/wbem/cim-html/`

de même nature (différents champs adresse). Le format propose quelques dictionnaires. Toutefois, il est souvent possible de spécifier des valeurs « libres » dans un champ lié à un champ restreint par un dictionnaire. Par exemple, le champ `SecurityIndication.MoreSpecificResources` complète le champ `SecurityIndication.Resources`, dont les valeurs sont restreintes par un dictionnaire.

Extensibilité CIM permet d'étendre le schéma standard par héritage.

XDAS/CADF XDAS est un standard défini dans les années 80 par des acteurs du monde UNIX réunis dans un groupe de travail de l'Open Group [16], un consortium de normalisation neutre et indépendant. Peu d'implémentations de ce standard ont vu le jour mais en 2007 le groupe de travail a été reformé, notamment à l'initiative de Novell [17]. Une implémentation open-source a alors été proposée : openXDAS [18]. Le groupe envisage visiblement la publication d'une nouvelle version mais, à la date de rédaction de ce présent document, cette nouvelle version n'est pas disponible sur le site de l'Open Group. Il semble que l'activité de Novell concernée par ce travail ait été cédée à NetIQ [19]. Récemment, le groupe de travail a initié une collaboration avec le DMTF pour intégrer XDAS V2 aux standards CADF et CIM du DMTF [20]. Les personnes concernées ont visiblement également participé au groupe de travail CEE. Actuellement, le DMTF a publié une version stable du format CADF qui reprend les principes communs aux formats de la « famille » XDAS. Il s'agit à l'heure actuelle du format le plus abouti et le plus actif de cette famille. C'est donc cette version du format que nous avons considérée dans l'étude détaillée des champs des différents formats.

Au départ, le format est dédié à la génération et au filtrage d'événements en général au niveau du système d'exploitation (UNIX). En pratique, il est plutôt adapté pour des événements provenant d'un système d'exploitation, en particulier ceux liés à l'authentification. La première version définit un schéma de manière succincte, une taxonomie d'événements ainsi qu'une API. La version fournie par NetIQ se concentre sur la définition du schéma et de la taxonomie correspondante. Le format CADF du DMTF est dédié à l'audit en général mais dans le contexte particulier du Cloud.

Références Les spécifications de la version originelle (XDAS V1) du format XDAS sont disponibles publiquement sur le site de l'Open Group [21]. NetIQ do-

16. `http://www.opengroup.org/`
17. `https://collaboration.opengroup.org/projects/security/xdas/`
18. `http://openxdas.sourceforge.net/`
19. `https://www.netiq.com/documentation/idm401/idm_sentinel/data/bqxvslh.html`
20. `http://www.opengroup.org/node/3037`
21. `https://www2.opengroup.org/ogsys/catalog/P441`

cumente la version de XDAS utilisée dans ses produits [22] (qui semble préfigurer le format XDAS V2). Le DMTF a publié une version stable de CADF sous la forme d'un document de spécification (DSP0262) sous format pdf. Cette spécification est complétée par d'autres documents qui illustrent l'implémentation de CADF au sein d'OpenStack. L'ensemble des documents est disponible sur le site Internet du DMTF [23].

Les différentes versions de XDAS reprennent des concepts communs, notamment le triplet « originator/observer », « initiator » et « target » qui correspondent peu ou prou aux concepts de « sonde », de « source » et de « cible » d'IDMEF. Toutefois, des différences notables apparaissent dans la définition des champs de chacune de ces classes suivant les formats.

La documentation du format XDAS V1 est très succincte (les champs ne sont pas toujours identifiés ni nommés clairement). La documentation fournie par NetIQ est succincte mais tous les champs y sont clairement identifiés. Toutefois, la sémantique de chaque champ est décrite très brièvement. En outre, la composition des champs agrégés n'apparaît clairement que sur la description de l'encodage en JSON. Le document de spécification de CADF est très complet (183 pages). Il précise clairement les objectifs et les concepts de ce format. La signification de chaque champ est explicitée ainsi que la philosophie générale du format, notamment à travers différents exemples. La spécification décrit en outre les types primitifs (entier, date, chaîne de caractères, etc.) et structurés qui sont ensuite utilisés pour spécifier les types des différents champs.

Transport et encodage A priori XDAS devrait être indépendant du transport et de l'encodage. Toutefois, NetIQ fournit un schéma JSON. Les exemples fournis laissent à penser que Syslog est utilisé comme transport. La spécification de CADF n'impose ni le transport ni l'encodage. Elle fournit des règles pour l'encodage en XML et en JSON, ainsi que des exemples utilisant ces deux encodages pour chaque champ du format.

Pouvoir d'expressivité Il est difficile d'évaluer précisément le pouvoir d'expressivité de XDAS en raison des différences entre les versions. Les concepts commun permettent d'exprimer des informations relatives à la sonde, la source et la cible. Toutefois, la version XDAS V1 et la version documentée par NetIQ proposent un nombre limité de champs pour chacune de ces classes. CADF offre un plus grand nombre de champs. Les formats XDAS et CADF ne proposent pas de champ permettant d'exprimer des informations relatives aux processus ou aux fichiers.

Structuration XDAS et CADF sont des formats relativement structurés. Ils s'appuient essentiellement sur trois classes qui elles-même comprennent des champs agrégés génériques (Entity, Account...). Toutefois, le nombre de champs est limité : CADF propose 48 champs. La version initiale du format propose une

22. `https://www.netiq.com/documentation/edir88/edirxdas_admin/data/bqppfzw.html`
23. `https://www.dmtf.org/standards/cadf`

taxonomie des événements (un dictionnaire de « catégories » d'événements) reprise par la version de NetIQ. Toutefois, ces catégories sont plutôt orientées « authentification ». CADF définit trois types de taxonomies : *Resource*, *Action* et *Outcome*. La première permet de décrire une ressource (associée à la source, la sonde ou la cible) sous la forme d'un arbre spécifié par une URI. Action permet d'exprimer le type d'activité décrite dans l'alerte sous une forme hiérarchique similaire à la précédente. Outcome permet de spécifier le résultat de cette action. Il s'agit d'un simple dictionnaire beaucoup plus limité que les deux précédents.

Extensibilité La documentation de NetIQ précise que n'importe quel champ additionnel peut-être ajouté dans le format pour compléter les champs standards. La spécification de CADF précise que le format peut-être étendu en publiant des profils qui respectent les règles principales de la spécification.

3.2 Synthèse des résultats

Le tableau 1 présente les résultats de l'analyse détaillée de chaque champ des différents formats.

Table 1. Richesse relative des formats

Format	IDMEF	CEF	LEEF	CIM	CEE	CADF
Nombre de champs	166	117	50	58	56	48
Nombre de champs normalisés	259	84	49	48	49	76
Nombre de champs traduisibles	259	65	20	29	39	72
Nombre de champs non traduisibles mais pertinents	0	15	11	11	5	3
Nombre de champs non traduisibles et peu pertinents	0	4	18	8	5	1
Nombre de champs pertinents	248	80	31	40	44	75
Couverture du format IDMEF	100 %	25 %	8 %	11 %	15 %	28 %
Richesse relative par rapport à IDMEF	100 %	32 %	13 %	16 %	18 %	30 %

La deuxième ligne de ce tableau comptabilise le **nombre de champs** proposés par la spécification de chaque format. Toutefois, lorsque nous souhaitons comparer ces champs, il apparaît assez rapidement que la granularité n'est pas toujours la même selon les formats. Ainsi, certains formats comme IDMEF ou CADF proposent des constructions permettant de stocker différentes informations à l'aide d'un nombre plus restreint de champs que les autres formats. A l'inverse, certains formats utilisent des champs distincts pour stocker des informations qui sont regroupées dans un seul champ par d'autres formats. Pour réaliser une étude comparative quantitative, nous avons calculé un **nombre de champs normalisés** (ligne 3) pour chaque format. Ce calcul s'appuie sur une représentation canonique arbitraire. Nous avons ensuite augmenté le nombre de champs pour les formats qui, par rapport à cette représentation canonique, stockent l'information dans un nombre plus restreint de champs. A l'inverse, nous avons diminué le nombre de champs des formats qui disséminent la même information dans des champs distincts. Cette métrique permet de mesurer quantitativement la richesse des différents formats de manière homogène. Cette première étape fait apparaître clairement que le format IDMEF est celui qui propose le plus grand nombre de champs.

Pour chaque format, nous avons analysé l'ensemble des champs et nous avons tenté de les traduire dans le format IDMEF, qui paraît le plus complet. Nous avons ainsi comptabilisé le nombre de champs, après normalisation, qu'il était possible de traduire de la sorte (**nombre de champs traduisibles**, ligne 4). Cela nous permet de calculer la **couverture du format IDMEF** (ligne 8). Nous définissons cette métrique comme le ratio du nombre de champs qu'il est possible de traduire par rapport au nombre total de champs d'IDMEF, après normalisation.

Pour les champs impossibles à traduire, deux cas de figure se présentent. Le format peut proposer des champs pertinents non prévus par IDMEF (**nombre de champs non traduisibles mais pertinents**, ligne 5). Ces champs constituent des pistes intéressantes pour étendre ou mettre à jour le format IDMEF. Nous avons également identifié des champs dont la sémantique n'était pas claire, qui sont très dépendants d'une solution ou dont l'intérêt pour le domaine des formats d'alertes de sécurité n'est pas évident (**nombre de champs non traduisibles et peu pertinents**, ligne 6). Nous avons également identifié le **nombre de champs pertinents d'un format** (qui peuvent être traduits en IDMEF ou non). Pour IDMEF, nous avons soustrait le nombre les champs que nous proposons de supprimer ou de rendre obsolètes (cf. section 4). Ces résultats nous permettent de calculer la **richesse relative par rapport à IDMEF**. Nous définissons cette métrique comme le ratio entre le nombre de champs pertinents du format par rapport à celui d'IDMEF.

Ces résultats confirment qu'IDMEF présente, de loin, l'expressivité la plus importante même s'il existe des champs proposés par d'autres formats qui ne peuvent être exprimés en IDMEF.

Le tableau 2 synthétise les résultats de l'analyse comparative des différents formats selon les critères définis en section 2.2. Comme décrit précédemment,

Table 2. Synthèse

Format	CEF	LEEF	IDMEF	CIM (sec)	XDAS (CADF)	CEE
Origine	HP	IBM	IETF (RFC 4765)	DMTF	The Open Group & DMTF	MITRE
Expressivité	++	−	+++	+	++	+?
Structuration	−−	−−	+++	+	++	−
Transport	Syslog	Syslog	IDXP	HTML	Syslog	Syslog
Encodage	Syslog + clé / valeur	Syslog + clé / valeur	XML	HTML	JSON XML	JSON XML

IDMEF se démarque clairement des autres formats en ce qui concerne l'expressivité. CEF et, dans une moindre mesure, CADF, sont également des formats très expressifs. CEF couvre la plupart des catégories proposées par IDMEF et il propose un certain nombre de champs pertinents qui ne sont pas présents dans IDMEF. Les autres formats sont en retrait. Toutefois, tous les formats étudiés permettent d'exprimer les champs les plus courants et les plus utiles dans le domaine (adresses de la source, de la cible, message, description de la sonde, etc.)

Concernant la structuration, les formats se rangent en deux catégories bien distinctes. La première catégorie correspond aux format peu structurés qui, tel CEF, on adopté une structure « à plat » sous la forme d'un ensemble de couples clé/valeur. A l'inverse, les formats très structurés comme IDMEF ou XDAS utilisent des classes agrégées. Cette solution permet de regrouper les champs selon le type d'information qu'ils sont censés décrire. Les défenseurs d'une solution « à plat » plaident pour une plus grande simplicité. Toutefois, cela reste vrai uniquement si le nombre de champs reste faible. Ceci est en contradiction avec le critère précédent : l'expressivité suppose un nombre de champs important. Dès lors, la forme structurée facilite la compréhension. On peut également noter qu'un format structuré peut facilement être transformé sous une forme clé/valeur : il suffit d'encoder la clé en concaténant les différentes classes agrégées jusqu'au champ souhaité.

Concernant le transport et l'encodage, les formats s'appuient principalement sur des encodages textuels classiques (XML, JSON) et des protocoles standards. En outre, tous les formats sont en principes indépendants de l'encodage et du transport et pourraient a priori utiliser d'autres encodage/transport que ceux préconisés. C'est particulièrement vrai pour les formats issus de standard pour

lesquels le transport et l'encodage ne sont généralement proposés qu'à titre d'exemple (la spécification de ces formats n'impose aucun encodage ou transport particulier).

4 Pistes d'améliorations de l'existant

IDMEF apparaît comme le format le plus adéquat pour l'échange d'alertes. Ce résultat n'est guère surprenant puisqu'il s'agit d'un format dédié à ce besoin. Toutefois, l'étude fait apparaître quelques lacunes que nous envisageons de corriger en proposant de faire évoluer le format. Nous pouvons distinguer trois types de lacunes :

- la complexité du format et de sa documentation ;
- l'absence de mise-à-jour ;
- l'absence de certaines informations qui sont proposées par d'autres formats ;
- l'absence d'implémentation supportant différents types de transports et d'encodages.

En pratique, IDMEF n'est pas si compliqué qu'il n'y parait à premier abord. Il est ainsi relativement aisé de construire une alerte avec les champs les plus courants, IDMEF n'imposant que très peu de champs obligatoires. Toutefois, la grande richesse du format peut dérouter le néophyte. La forme de la RFC peut également constituer un frein. Il convient donc selon nous de compléter la documentation du format par des tutoriaux permettant d'illustrer en pratique l'utilisation d'IDMEF sur différents cas de figure représentatifs des différentes catégories de sondes couramment utilisées de nos jours (NIDS, firewall, Anti-Virus, WAF, etc.). En outre, nous avons réalisé des diagrammes UML permettant de mieux visualiser la structure du format, notamment via l'utilisation de couleurs pour différencier les différentes catégories de classes. Ces diagrammes permettent à la fois d'avoir une vue d'ensemble et de zoomer sur une sous-partie afin d'identifier, par exemple, l'ensemble des champs possibles pour une classe donnée.

IDMEF souffre également de son âge et de l'absence de mise-à-jour. Ainsi, le format reflète les problématiques qui étaient d'actualité lors de sa création. Certains aspects peuvent aujourd'hui apparaître comme moins cruciaux tandis que d'autre sont sous-représentés (par exemple, les services Web). A l'inverse, l'analyse des formats concurrents nous a permis d'identifier un certain nombre de champs qui ne sont pas présents dans IDMEF mais qui semblent pertinents.

Enfin, il nous paraît important de proposer une implémentation de référence, sous la forme d'une bibliothèque utilisable dans différents langages, supportant différents types de transport et d'encodage. En effet, à l'heure actuelle, peu de produits implémentent IDMEF. Il n'existe pas d'implémentation générique (i.e. qui ne soit pas spécifique à un produit, notamment en termes de transport et d'encodage) et utilisable par tous. Cette limitation n'est pas propre à IDMEF mais il nous parait essentiel de fournir une telle implémentation pour facili-

ter l'adhésion. Le développement de cette bibliothèque, en cours de réalisation, constitue l'un des objectifs du projet SECEF.

Nous présentons par la suite certaines pistes d'amélioration que nous envisageons. Ces pistes seront analysées dans la suite du projet SECEF afin de proposer des évolutions du format IDMEF. Les travaux sur ce sujet sont en cours de réalisation au sein du projet.

Nous avons identifiés quatre types d'évolution :

— l'ajout de champs dans les classes existantes ;
— la suppression (ou la déclaration obsolète) de certaines classes ou mécanismes proposés dans IDMEF V1 ;
— l'ajout de classes ;
— la mise à jour des dictionnaires ;

Nous détaillons par la suite ces différents points d'amélioration.

4.1 Ajouts de champs dans les classes existantes

L'étude des différents formats nous a permis d'identifier certains champs manquant dans IDMEF.

Identification de la fin d'une attaque IDMEF propose plusieurs champs d'horodatage pour spécifier l'observation d'un événement ou l'émission de l'alerte. Ces champs sont typiquement associés au début d'un événement. Il serait intéressant de pouvoir spécifier la fin d'un événement, notamment pour les attaques telles qu'un scan de port. Ce type de champs est proposé par CEF et certaines versions d'XDAS.

Identification du protocole de transport IDMEF permet d'identifier le protocole et le port associé à un service. Toutefois, ces champs permettent a priori de spécifier le protocole de plus haut niveau qu'il soit possible d'identifier. Dans beaucoup de cas, il s'agit d'un protocole applicatif comme HTTP ou DNS. Il parait intéressant de pouvoir également préciser le protocole de transport utilisé, notamment pour des protocoles applicatifs qui peuvent utiliser différents protocoles de transport (par exemple DNS). Les formats CEF, LEEF et CIM proposent un champ distinct pour spécifier ce type d'information.

Identification des interfaces d'entrées et de sortie IDMEF permet de spécifier une interface réseau pour la source et la cible d'une alerte. Toutefois, le format ne permet pas de spécifier d'interface pour l'analyseur. En outre, il n'est pas possible de distinguer l'interface d'entrée de l'interface de sortie empruntées par un paquet réseau. Ce type d'information paraît pertinent pour les alertes remontées par des sondes réseau qui possèdent de multiples interfaces tels les routeurs, pare-feux ou NIPS/NIDS. CEF propose des champs permettant d'identifier l'interface d'entrée et de sortie de la sonde à l'origine de l'alerte.

Gestion explicites des données de géolocalisation IDMEF permet de spécifier la géolocalisation d'un noeud, qu'il s'agisse de la source, de la cible ou de l'analyseur. Toutefois, cette information est contenue dans un unique champ dont la sémantique est vague (il s'agit d'une chaîne de caractère). Suivant les implémentations et les déploiements, la structuration des données stockées dans ce champ peut varier, ce qui constitue une limitation pour les traitements automatiques. Il parait nécessaire de préciser le format de ce champ et/ou de proposer des champs additionnels distincts permettant d'identifier les données de géolocalisation (longitude/latitude/altitude, coordonnées GPS, etc.). Le format CADF propose de tels champs.

Attachement du log originel IDMEF permet d'attacher des données issues des journaux à l'aide du champs `AdditionnalData`. Toutefois, ce champ n'est pas dédié à cet usage et permet d'étendre le format de manière général. Il parait intéressant de proposer un champ dédié permettant d'embarquer le log originel ayant permis de créer une alerte, comme le proposent CEE et XDAS.

Identification du thread IDMEF permet d'identifier le numéro de processus (PID) mais pas le numéro de thread. Une modification mineur consiste à ajouter un champ dédié pour cette information comme le propose le format CEE.

Identification de la catégorie d'un noeud IDMEF permet de spécifier la catégorie d'un analyseur (champ `Analyzer.Class`). Cela permet par exemple de regrouper des équipements différents (nom ou vendeur différents) offrant la même fonctionnalité (par exemple NIDS, parefeux, HIDS, etc.). Toutefois, IDMEF ne permet pas d'associer une catégorie aux autres noeuds (la source et la destination d'une alerte). Il parait intéressant d'ajouter des champs similaires aux classes `Alert.Source` et `Alert.Target` ou de déplacer ce champ dans la classe `Node` afin de pouvoir associer une catégorie à chaque noeud (par exemple serveur, poste client, routeur, etc.).

Compteur d'occurence IDMEF permet de spécifier des alertes de corrélation qui peuvent agréger différentes alertes primaires. Pour cela, IDMEF propose un champ qui contient l'identifiant de toutes les alertes agrégées. Cela suppose que ces alertes aient été transmises. Parfois, il peut être nécessaire d'agréger un nombre important d'alertes très similaires afin de limiter le nombre d'alertes transmises. Dans ce cas de figure, il parait intéressant d'indiquer le nombre d'alertes agrégées dans un champ dédié de la classe `CorrelationALert`. Un tel champ est proposé par les formats CEF, CIM et XDAS.

Domaine d'authentification IDMEF permet de spécifier les identifiants associés à un utilisateur. Toutefois, ces identifiants n'ont de sens qu'au sein d'un « domaine d'identification ». Ce dernier peut correspondre à un noeud réseau

dans le cas de comptes locaux. Il peut également s'agir d'un serveur LDAP ou d'un domaine ActiveDirectory. Il parait intéressant d'ajouter un champ dédié associé à chaque identifiant afin de pouvoir préciser son domaine. LEEF, CEE et XDAS proposent un tel champ.

Catégorie d'utilisateur IDMEF permet de spécifier les privilèges associés à un utilisateur via le champ `User.UserID`. Toutefois, il s'agit d'une vision très « bas niveau ». Il paraît intéressant de pouvoir spécifier une catégorie d'utilisateur (par exemple administrateur, utilisateur authentifié, utilisateur anonyme, etc.). CEF, LEEF et XDAS proposent des champs dédiés à cet usage.

4.2 Obsolescence de certains mécanismes d'IDMEF

L'analyse détaillée du format IDMEF et le retour d'expérience lié à son utilisation permettent d'identifier certains points qui sont peu utilisés ou paraissent peu pertinents aujourd'hui. Nous proposons de supprimer ces point dans la future version du format ou du moins de les déclarer obsolètes pour garantir une compatibilité descendante.

Ainsi IDMEF propose un mécanisme d'identification unique des classes via le champ `ident`. Ce mécanisme permet d'associer un identifiant unique à chaque instance de classe. Une instance de classe est définie par un ensemble de valeur identique pour les différents champs qui la composent. L'objectif de ce mécanisme était de compresser les messages en évitant de répéter des informations déjà spécifiées au préalable mais la documentation de la RFC est peu claire à ce sujet. En pratique, il n'est pas utilisé dans les implémentations. En outre, il apparaît plus pertinent de gérer la compression au niveau de l'encodage et du transport, en utilisant des mécanismes standards. Par exemple, il est possible d'utiliser différentes techniques de compression des messages XML (gzip, XGrind, XMill, etc.). Nous proposons de déclarer ces champs obsolètes ou de préciser leur utilisation dans un cadre plus actuel. Ainsi, cet identifiant pourrait servir de « clé externe » pour référencer des objets d'un autre format, par exemple les formats de description de la topologie.

IDMEF propose la classe `OverflowAlert` qui hérite de la classe `Alert`. L'idée était de spécialiser les alertes suivant le type d'attaque. Toutefois, il s'agit de la seule classe fille proposée par le format dans ce but. Cela pose un problème d'homogénéité et de représentativité des attaques modernes. En outre, les informations supplémentaires qu'elle permet de spécifier sont très précises et spécifiques (taille et nom du buffer). En pratique, peu de sondes sont capables de renseigner ces champs et l'intérêt de ce type d'information pour un traitement automatisé des alertes n'est pas évident. Nous proposons de supprimer cette classe ou de la déclarer obsolète.

4.3 Ajout de classes

IDMEF propose deux classes héritants de la classe `Service` : `WebService` et `SNMPService`. Il s'agit, comme dans le cas de la classe `OverflowAlert` de

spécialiser la classe mère en fonction du type de service. En l'état, le nombre de classes filles proposées n'est pas suffisant pour couvrir les services les plus courants. Cela traduit encore une fois un problème d'homogénéité. Toutefois, il parait nécessaire de compléter le schéma car les différents services nécessitent de spécifier des informations différentes. Nous proposons donc d'ajouter des classes pour couvrir les services les plus courants (par exemple LDAP et SIP). En outre, la classe `WebService` mérite d'être étoffée. En particulier, il parait intéressant d'ajouter des champs dédiés permettant de spécifier des paramètres tels que host, referer ou cookie.

4.4 Mise à jour des dictionnaires

IDMEF utilise des dictionnaires pour contraindre le type de certains champs. Cette pratique est louable car elle permet d'homogénéiser les valeurs. Toutefois, elle pose le problème de la mise-à-jour de ces dictionnaires afin de tenir compte des nouveaux usages et de l'évolution de l'éco-système en général. Actuellement, ces dictionnaires sont définis dans la spécification du format. Pour faciliter leur mise à jour, nous envisageons de les extraire et de les gérer indépendamment, par exemple en s'appuyant sur l'IANA pour maintenir les énumérations.

En pratique, les mises-à-jour devraient notamment porter sur les dictionnaires associés aux champs suivants :

- `Impact.Type` ;
- `Action.Category` ;
- `Reference.Origin` ;
- `Node.Category` ;
- `File.fsType` ;
- `Checksum.Algorithm` ;
- `Address.Category`.

En particulier, nous envisageons d'étendre ce dernier dictionnaire afin de pouvoir spécifier des adresses translatées (utilisation du NAT).

5 Conclusion

Nous avons présenté dans cet article une étude comparative de différents formats d'alertes réalisée dans le cadre du projet SECEF. Les résultats de cette étude montrent que tous les formats permettent de spécifier les champs les plus couramment utilisés lors de la génération d'alerte. En outre, ils sont tous relativement indépendants de l'encodage et du transport, même si la spécification de certains formats s'appuie sur un encodage ou un transport particulier, essentiellement à des fins d'illustration. Toutefois, des différences significatives existent concernant l'expressivité et la structuration de ces formats. Les résultats font clairement apparaître la supériorité d'IDMEF dans ce domaine, ce qui n'est guère surprenant car il a été développé spécifiquement pour l'échange d'alertes de sécurité, ce qui n'est pas le cas des autres formats. Il s'agit du format le

plus structuré et de celui qui offre la plus grande richesse en termes d'expressivité. Toutefois, cette étude a également permis d'identifier quelques lacunes d'IDMEF, notamment au regard des possibilités offertes par les autres formats. Ces résultats permettent d'identifier des pistes d'amélioration visant à créer de nouvelles classes, de nouveau champs mais également de gérer l'obsolescence de certains mécanismes d'IDMEF. Enfin, il est selon nous important de pouvoir mettre à jour les dictionnaires, ce qui nécessite peut-être de les dissocier de la spécification du format afin de pouvoir les faire évoluer plus fréquemment.

Les résultats de cette étude vont ainsi permettre au consortium de proposer une mise à jour du format IDMEF prenant en compte les pistes d'évolution identifiées. Le travail en cours consiste à valider les points qui feront l'objet d'une mise à jour et la forme adoptée (ajout/suppression de champs, de classe, etc.). En outre, l'étude comparative détaillée des différents champs nous a permis de constituer une table de conversion entre les différents formats et IDMEF. Cette table pourra servir de référence pour l'implémentation de passerelles entre les formats.

La présente étude s'est concentrée sur le format des messages (schéma et typage). Les problématiques d'encodage et de transport, bien qu'orthogonales aux problèmes évoqués dans cette étude, sont également d'une importance cruciale pour l'implémentation d'une solution efficace. Nous pensons que l'adoption d'un standard passe en grande partie par la mise à disposition d'outils de référence permettant d'utiliser ce standard. Un des objectifs futur du projet SECEF consiste donc à mettre à disposition une bibliothèque permettant d'échanger des messages IDMEF dans différents langages. Nous envisageons, pour cette bibliothèque, de supporter différentes forme d'encodage et de transport.

Références

1. Debar, H., Curry, D., Feinstein, B. : The Intrusion Detection Message Exchange Format (IDMEF). RFC 4765 (Experimental) (March 2007), http://www.ietf.org/rfc/rfc4765.txt
2. Hollnagel, E., Paries, J., Woods David, D., Wreathall, J. : Resilience engineering in practice : A Guidebook. Ashgate Studies in Resilience Engineering, Ashgate Publishing (Dec 2010), https://hal-mines-paristech.archives-ouvertes.fr/hal-00613345
3. Pawliński, P., Jaroszewski, P., Urbanowicz, J., Jacewicz, P., Zielony, P., Kijewski, P., Gorzelak, K. : Standards and tools for exchange and processing of actionable information. Tech. Rep. TP-04-14-999-EN-N, ENISA (November 2014), https://www.enisa.europa.eu/activities/cert/support/actionable-information/standards-and-tools-for-exchange-and-processing-of-actionable-information