# RFC 8274 : Incident Object Description Exchange Format Usage Guidance

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 20 novembre 2017

Date de publication du RFC : Novembre 2017

https://www.bortzmeyer.org/8274.html

---

Le format IODEF, dont la dernière version est décrite dans le RFC 7970 [1], est un format structuré permettant l'échange de données sur des incidents de sécurité. Cela permet, par exemple, aux CSIRT de se transmettre des données automatiquement exploitables. Ces données peuvent être produites automatiquement (par exemple par un IDS, ou bien issues du remplissage manuel d'un formulaire). IODEF est riche, très riche, peut-être trop riche (certaines classes qu'il définit ne sont que rarement utilisées). Il peut donc être difficile de programmer des outils IODEF, et de les utiliser. (En pratique, il me semble qu'IODEF est peu utilisé.) Ce RFC, officiellement, est donc chargé d'aider ces professionnels, en expliquant les cas les plus courants et les plus importants, et en guidant programmeurs et utilisateurs.

Personnellement, je ne suis pas convaincu du résultat, ce RFC me semble plutôt un pot-pourri de diverses choses qui n'avaient pas été mises dans la norme.

La section 3 du RFC discute de l'utilisation de base d'IODEF. Reprenant la section 7.1 du RFC 7970, elle présente le document IODEF minimum, celui avec uniquement l'information obligatoire :

```
<?xml version="1.0" encoding="UTF-8"?>
<IODEF-Document version="2.00" xml:lang="en"
      xmlns="urn:ietf:params:xml:ns:iodef-2.0"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="http://www.iana.org/assignments/xml-registry/schema/iodef-2.0.xsd">
   <Incident purpose="reporting" restriction="private">
       <IncidentID name="csirt.example.com">492382</IncidentID>
```

1. Pour voir le RFC de numéro NNN, https://www.ietf.org/rfc/rfcNNN.txt, par exemple https://www.ietf.org/rfc/rfc7970.txt

```
      <GenerationTime>2015-07-18T09:00:00-05:00</GenerationTime>
      <Contact type="organization" role="creator">
        <Email>
          <EmailTo>contact@csirt.example.com</EmailTo> <!-- Pas
   réellement obligatoire, mais le document serait vraiment
   sans intérêt sans lui. -->
        </Email>
      </Contact>
    </Incident>
</IODEF-Document>
```

Un tel document, comportant une instance de la classe `Incident`, qui comprend elle-même une instance de la classe `Contact`, serait syntaxiquement correct mais n'aurait guère d'intérêt pratique. Des documents un peu plus réalistes figurent dans l'annexe B.

Le programmeur qui génère ou traite des fichiers IODEF n'a pas forcément à mettre en œuvre la totalité des classes. Il peut se contenter de ce qui est important pour son ou ses scénarios d'utilisation. Par exemple, si on travaille sur les dDoS, la classe `Flow` est la plus importante, puisqu'elle décrit le trafic de l'attaque. (L'annexe B.2 du RFC contient un fichier IODEF décrivant une attaque faite avec LOIC. Je l'ai copié ici dans (en ligne sur `https://www.bortzmeyer.org/files/ddos-iodef.xml`).) De même, si on travaille sur le C&C d'un logiciel malveillant, les classes `Flow` et `ServiceName` sont cruciales. Bref, il faut analyser ce dont on a besoin.

La section 4 du RFC mentionne les extensions à IODEF. Si riche que soit ce format, on peut toujours avoir besoin d'autres informations et c'est pour cela qu'IODEF est extensible. Par exemple, le RFC 5901 décrit une extension à IODEF pour signaler des cas de hameçonnage. Évidemment, on ne doit définir une extension que s'il n'existe pas de moyen existant de stocker l'information dans de l'IODEF standard.

La section 4 rappelle aussi aux développeurs que, certes, IODEF bénéfice d'un mécanisme d'indication de la confidentialité (l'attribut `restriction`, qui se trouve dans les deux exemples que j'ai cité), mais qu'IODEF ne fournit aucun moyen technique de le faire respecter. Les documents IODEF étant souvent sensibles, puisqu'ils parlent de problèmes de sécurité, le programmeur qui réalise un système de traitement de fichiers IODEF doit donc mettre en œuvre des mesures pratiques de protection de la confidentialité (chiffrement des fichiers stockés, par exemple).

Questions mise en œuvre d'IODEF, le RFC 8134 détaille des programmes existants, indique où les récupérer quand ils sont accessibles en ligne, et analyse leurs caractéristiques. C'est le cas par exemple d'iodeflib <`http://www.decalage.info/python/iodeflib`>.