Vulnérabilité affectant la bibliothèque PHP dompdf

RAPPORT D'ANALYSE

24/08/2022

Version : 1.0.0
Date d'enregistrement: 24/08/2022
Nombre de pages : 10

Table des matières

1 Contexte

Le 16 mars 2022, des chercheurs ont publié des travaux concernant la découverte d'une vulnérabilité au sein de la bibliothèque PHP de rendu de PDF : *dompdf*. Il s'agit essentiellement d'un convertisseur HTML vers PDF qui est très largement utilisé dans le développement de sites PHP.

Le 24 mars 2022, l'éditeur a publié la version 1.2.1 de *dompdf* qui corrige cette vulnérabilité.

La vulnérabilité permet à un attaquant d'exécuter du code arbitraire à distance lorsque la bibliothèque *dompdf* est utilisée par une application pour convertir du contenu qui serait partiellement fourni par l'utilisateur. L'attaque se déroule ainsi :

- Un attaquant, en mesure de transmettre du contenu à convertir, peut inciter l'application à télécharger des polices de caractères true type font (*ttf*) à distance [1]. Lors de la conversion au format PDF, la bibliothèque procédera au téléchargement du fichier de police de caractères *ttf* présent sur le serveur de l'attaquant [2]. Cette possibilité de téléchargement, combinée à deux défauts de contrôle de la bibliothèque, permet à l'attaquant d'injecter du code PHP sur le serveur de l'application. En effet :
  - Pour s'assurer que le format de fichier correspond bien à une police *ttf*, *dompdf* fait appel à la bibliothèque php-font qui n'effectue qu'une vérification partielle du fichier. L'attaquant est donc en mesure de préparer une police de caractères à télécharger depuis Internet sous forme de fichier *ttf* mais contenant du code PHP malveillant ;
  - Par ailleurs, lors du téléchargement de la police par *dompdf*, l'extension du fichier de police n'est pas vérifiée. Ainsi, un fichier de type *ttf* mais portant l'extension *.php* peut donc être déposé dans le dossier de réception accessible depuis Internet.
- Les fichiers ayant l'extension .php étant exécutés par le serveur web, il suffit à l'attaquant d'effectuer une requête directe de type « *GET* » sur ce fichier pour que celui-ci provoque une exécution de code arbitraire à distance.

Toute application utilisant la bibliothèque *dompdf* pour la conversion de contenu utilisateur au format PDF doit impérativement utiliser une version à jour de cette bibliothèque. Par ailleurs, les recommandations suivantes permettent d'apporter des éléments de sécurisation supplémentaires destinés à prévenir de vulnérabilités similaires liées aux fonctionnalités de téléversement de fichiers sur un serveur.

1. Cela est possible en insérant des règles CSS font-face dans les données HTML qu'il soumet à l'application.
2. La fonctionnalité de téléchargement de police de caractère est possible sur les versions ≤ 0.8.5 de la bibliothèque. Dans les versions ultérieures, il faut activer l'option $isRemoteEnabled de *dompdf*.

2 Versions affectées

La vulnérabilité a été découverte officiellement en 2021, cependant celle-ci avait fait l'objet d'une première déclaration d'anomalie en 2019 [3], sans toutefois être considérée comme une vulnérabilité. Les échanges concernant cette anomalie mentionnent notamment qu'elle aurait pu être introduite dans la version 0.8.3, mais sans certitude. Le code incriminé est présent au moins depuis la version 0.7.0 même si certains éléments permettant d'exploiter la vulnérabilité selon la méthode décrite dans le présent document ne sont pas présents. Il est donc prudent de considérer que toutes les versions antérieures à la version 1.2.1 sont vulnérables.

3 Sécurisation des applications utilisant *dompdf*

3.1 Recommandations proposées par l'éditeur

1. Installer la bibliothèque en dehors de la racine du serveur web (document root) de façon à ne pas donner accès aux fichiers *dompdf* depuis Internet ;
2. Supprimer tout le code qui ne concerne pas la bibliothèque, en particulier certains tests unitaires fournis par *dompdf* et ajoutés après la version 0.7.0 ;
3. Restreindre l'accès à *dompdf* : si la première recommandation ne peut pas être respectée, il est conseillé de :
    a) Limiter les accès à la bibliothèque aux seuls hôtes et adresses IP autorisées ;
    b) Mettre en place un mécanisme d'authentification
4. Désactiver le support des scripts embarqués, se référer au lien éditeur [4] pour un exemple.

Référence :

- `https://github.com/dompdf/dompdf/wiki/Securing-dompdf`

3. `https://github.com/dompdf/dompdf/issues/1976`
4. `https://github.com/dompdf/dompdf/wiki/Securing-dompdf#4-disable-embedded-script-support`

3.2 Recommandations proposées par l'ANSSI

1. Mettre à jour *dompdf* vers une version récente (1.2.1 ou ultérieure) et désactiver l'option *$isRemoteEnabled* dans la configuration de la bibliothèque ;

2. Vérifier que le dossier d'installation de *dompdf* et donc, par corollaire, que son sous-dossier de téléversement de fichiers, ne soient pas accessibles directement depuis Internet ;

3. Vérifier également que les fichiers présents dans le répertoire de téléversement ne disposent pas de droits d'exécution. Il faut aussi s'assurer que le propriétaire du répertoire de téléversement de fichiers soit le même utilisateur que celui utilisé pour exécuter le service (généralement l'utilisateur *www-data* ou *apache*) et que les permissions soient définies sur 640.

    a) Corrections des droits et permissions associés aux fichiers présents dans le dossier de téléversement :

    ```
    chown apache:apache -R uploads/ && chmod 640 -R uploads/
    ```

    b) Mise en place d'ACLs spécifiques sur le dossier de téléversement avec setfacl :

    ```
    setfacl -m user:apache:rw,group:apache:rw uploads/
    ```

4. Configurer la directive "*allow_url_include = Off*" dans le fichier php.ini de façon à désactiver la gestion des URL dans les enveloppes PHP fopen ;

5. Désactiver l'interprétation par Apache des fichiers *.htaccess* ainsi que l'interprétation par le module PHP des fichiers contenant les extensions PHP les plus communes dans le dossier de téléversement de *dompdf*. Il est vivement *déconseillé* d'inclure les directives dans un fichier *.htaccess*, et en particulier si celui-ci est situé à la racine du dossier de téléversement. En effet, si des mesures de filtres par le développeur n'ont pas été préalablement effectuées, un attaquant serait en mesure de téléverser son propre fichier *.htaccess*. Cela aurait pour effet soit d'écraser celui d'origine soit par « effet de cascade » de lui permettre de bénéficier de la primauté sur l'original, modifiant ainsi le comportement et la sécurité globale du serveur web. En guise de prévention, une solution consiste en la modification du fichier de configuration d'Apache, dont le chemin par défaut est */etc/apache2/sites-enabled/monsite.fr* et d'y fournir les directives suivantes :

```
<Directory /var/www/html/mon_site/upload/>
     AllowOverride None
     Options None
     IndexIgnore *

     SetHandler none
     SetHandler default-handler

     php_flag engine off
     RemoveHandler .cgi .php .php3 .php4 .php5 .phtml .pl .py

    <Files *>
          AllowOverride None
          Options None
          IndexIgnore *

          SetHandler none
          SetHandler default-handler

          php_flag engine off
          RemoveHandler .cgi .php .php3 .php4 .php5 .phtml .pl .py
   </Files>
</Directory>
```

6. Mettre en place un mécanisme de pare-feu applicatif pour la fonction de téléversement. Quelques éléments de détection sont donnés à titre d'exemple. Ces éléments ne sont ni exhaustifs ni propres à la bibliothèque *dompdf* :

    a) La présence de code arbitraire dans des fichiers légitimement attendus par l'application [5] ;

    b) La présence de fichier *.htaccess* au sein de fichiers légitimes. Un attaquant, dans l'objectif de chaîner ses attaques, cherchera à modifier la configuration initiale d'Apache, en introduisant un fichier *.htaccess* avec des directives spécialement conçues. Par exemple, celui-ci cherchera à réactiver l'interprétation par le module PHP des fichiers qui auraient été téléversés dans le dossier idoine. La directive « *AddType application/x-httpd-php .jpg .evil* » permettrait ainsi l'interprétation par le module PHP de fichiers portant l'extension choisie par l'attaquant. En effet, *AddType* associe les extensions de noms de fichiers au type de contenu spécifié.

5. Cette règle peut être complexe à mettre en œuvre en fonction des capacités du pare-feu applicatif en matière de reconnaissance de codes scriptés ou compilés et les techniques d'obscurcissement et d'encodage employés par l'attaquant.

Plusieurs points sont néanmoins à prendre en compte :

- La directive *php_flag engine off* ne fonctionnera que si PHP est installé en tant que module Apache. Elle ne fonctionnera pas si PHP est exécuté en tant que CGI / FastCGI ;
- Les directives *SetHandler none*, *SetHandler default-handler* et *RemoveHandler* produisent globalement les mêmes effets. Il est probable que des configurations différentes de serveur nécessitent des paramètres différents ;
- La directive *AllowOverride None* force le fichier *.htaccess* à être complètement ignoré. Par conséquent, même si un fichier *.htaccess* se trouvait dans le répertoire (téléversé par un attaquant par exemple), il sera ignoré et ne pourra pas remplacer ou prendre la primauté sur les directives ci-dessus ;
- Cependant, il est important de comprendre que ces directives ne désactivent pas nécessairement l'exécution des scripts dans le dossier. Elles empêchent le client (agent utilisateur / navigateur) de réaliser des requêtes par adressage direct (HTTP) aux scripts présents dans un répertoire. Tout script peut toujours être exécuté dans ce répertoire s'il est "appelé" (par exemple, inclus) à partir d'un autre script mais qui serait présent en dehors du répertoire ciblé, en l'occurrence ici : */monsite/upload*. Une vulnérabilité de type LFI permettrait ainsi l'inclusion de fichiers bien que le dossier de téléversement soit situé en dehors de la racine du site et pour lequel l'interprétation des fichiers PHP présents a été désactivée. Ceci peut entraîner in fine l'interprétation par le module PHP de scripts présents dans des fichiers préalablement déposés sur le serveur. De manière générale, ceci vaut pour toute application offrant de telles fonctionnalités de téléversement de fichiers, et n'est donc pas spécifique au cas de *dompdf* ;

Les recommandations énoncées ne présument pas nécessairement d'une protection exhaustive face à ce type d'attaque. Des attaques de type « inclusion de fichier local » (Local File Inclusion, LFI) sont un autre vecteur permettant notamment d'exploiter cette vulnérabilité ;

Références Apache :

- `https://httpd.apache.org/docs/2.4/mod/core.html`
- `https://httpd.apache.org/docs/2.4/fr/handler.html`
- `https://httpd.apache.org/docs/current/fr/mod/mod_mime.html#addtype`

3.3 Autres mesures proposées par l'ANSSI

Dans le cadre du traitement de cette vulnérabilité, afin de détecter une possible compromission, il est recommandé de détecter le téléchargement de fichiers suspects ou des tentatives de connexions distantes suspectes. Le téléchargement d'une police se fait manifestement au travers du descripteur src :url, associé à la règle @font-face [6], qui permet de définir la ressource qui contient les données relatives à une police de caractères. Cependant cette recherche peut s'avérer difficile (encodage, obscurcissement de la requête, moyen détourné de type LFI, etc.).

En outre, il conviendra de vérifier la présence de code PHP au sein de fichiers ne portant pas l'extension .php* et de les supprimer. La présence des caractères de début d'un code PHP sont caractéristiques - « <? » - et peuvent aider à détecter la présence de code malveillant au sein de fichiers jugés inoffensifs. Cependant, cette recherche doit aussi prendre en compte d'éventuels mécanismes d'encodage [7] et d'obscurcissement utilisés par l'attaquant pour masquer ses charges utiles.

6. https://developer.mozilla.org/fr/docs/Web/CSS/@font-face/src

7. Par exemple, en UTF-8 un caractère est encodé sur un octet, mais en UTF-16 le même caractère sera encodé sur deux octets. Cela signifie que les caractères « <? » (3c3f) seront donc encodés en « .<.? » (003c003f).

4 Conclusion

Le cas de *dompdf* nous permet de mettre en évidence le fait que les bibliothèques qui font usage de fonctionnalités implicites ou explicites de téléversement de fichiers exposent les applications à des attaques qui peuvent être difficiles à identifier lorsque la sécurité d'une application est étudiée. Il est donc essentiel de mettre en place les bonnes pratiques de façon systématique, notamment les mesures non exhaustives suivantes :

- Toute application, qui traite un contenu transmis par un tiers, doit vérifier les entrées utilisateurs avant leur traitement, afin d'empêcher les tentatives d'injection de code arbitraire (SQL, HTML, CSS, etc.) ;
- Utiliser une version à jour des logiciels utilisés par l'application (PHP et Apache HTTP Server, etc.) ;
- Mettre en place un contrôle des flux sortants afin de bloquer les flux non explicitement autorisés ;
- Mettre en place des outils de collecte et d'analyse des journaux ;
- Mettre en place un mécanisme de pare-feu applicatif permettant de bloquer les requêtes utilisateurs non conformes ou présentant un contenu considéré comme malveillant, en fonction des capacités du pare-feu applicatif.

Le CERT-FR tient à rappeler que les attaquants peuvent chaîner l'exploitation de plusieurs vulnérabilités pour aboutir à une exploitation réussie. La sécurité d'une application repose donc sur une analyse des risques globale et le déploiement de mesures de protection formant un ensemble cohérent, respectant le principe de défense en profondeur.

—

La mise à jour d'un produit ou d'un logiciel est une opération délicate qui doit être menée avec prudence. Il est notamment recommandé d'effectuer des tests autant que possible, notamment afin de mesurer les effets de bord possibles, avant tout déploiement en production. Des dispositions doivent également être prises pour garantir la continuité de service en cas de difficultés lors de l'application des mises à jour comme des correctifs ou des changements de version.

Version 1.0.0 - 24/08/2022

AGENCE NATIONALE DE LA SÉCURITÉ DES SYSTÈMES D'INFORMATION

ANSSI - 51, boulevard de la Tour-Maubourg - 75700 PARIS 07 SP

www.ssi.gouv.fr