Équipe d'ingénierie de l'Internet (IETF)
Request for Comments : 6454
Catégorie : Sur la voie de la normalisation
ISSN: 2070-1721

A. Barth, Google, Inc.

décembre 2011
Traduction Claude Brière de L'Isle

Concept d'origine sur la Toile

Résumé
Le présent document définit le concept d'une "origine", qui est souvent utilisé comme portée d'une autorité ou privilège par les agents d'utilisateur. Normalement, les agents d'utilisateur isolent le contenu restitué de différentes origines pour empêcher des opérateurs de sites malveillants de la Toile d'interférer avec le fonctionnement des sites bénins de la Toile. En plus de souligner les principes qui sous-tendent le concept d'origine, le présent document détaille comment déterminer l'origine d'un URI et comment mettre en série une origine dans une chaîne. Il définit aussi un champ d'en-tête HTTP, nommé "Origin", qui indique quelles origines sont associées à une demande HTTP.

Statut de ce mémoire
Ceci est un document de l'Internet en cours de normalisation.

Le présent document a été produit par l'équipe d'ingénierie de l'Internet (IETF). Il représente le consensus de la communauté de l'IETF. Il a subi une révision publique et sa publication a été approuvée par le groupe de pilotage de l'ingénierie de l'Internet (IESG). Tous les documents approuvés par l'IESG ne sont pas candidats à devenir une norme de l'Internet ; voir la Section 2 de la RFC5741.

Les informations sur le statut actuel du présent document, tout errata, et comment fournir des réactions sur lui peuvent être obtenues à http://www.rfc-editor.org/info/rfc6454

Table des matières

1. Introduction

Les agents d'utilisateur interagissent avec les contenus créés par un grand nombre d'auteurs. Bien que beaucoup de ces auteurs soient bien intentionnés, certains peuvent être malveillants. Dans la mesure où les agents d'utilisateur entreprennent des actions fondées sur les contenus qu'ils traitent, les mises en œuvre d'agent d'utilisateur peuvent souhaiter restreindre la capacité des auteurs malveillants à perturber la confidentialité ou l'intégrité des autres contenus ou serveurs.

Par exemple, considérons un agent d'utilisateur HTTP qui rend des contenus HTML restitués de divers serveurs. Si l'agent d'utilisateur exécute les scripts contenus dans ces documents, la mise en œuvre d'agent utilisateur peut souhaiter empêcher les scripts restitués d'un serveur malveillant de lire les documents mémorisés sur un serveur honnête, qui par exemple, peut être derrière un pare-feu.

Traditionnellement, les agents d'utilisateur ont divisé les contenus en fonction de leur "origine". Plus précisément, les agents d'utilisateur permettent aux contenus restitués d'une origine d'interagir librement avec d'autres contenus restitués de cette origine, mais les agents d'utilisateur restreignent la façon dont ces contenus peuvent interagir avec des contenus provenant d'autres origines.

Le présent document décrit les principes qui se trouvent derrière ce qu'on appelle la politique de la même origine ainsi que les "écrous et boulons" de la comparaison et de la mise en série des origines. Le présent document ne décrit pas toutes les facettes de la politique de la même origine, dont les détails sont laissés à d'autres spécifications, comme HTML [HTML] et WebSockets [RFC6455], parce que les détails sont souvent spécifiques de l'application.

2. Conventions

2.1 Critères de conformité

Les mots clés "DOIT", "NE DOIT PAS", "EXIGE", "DEVRA", "NE DEVRA PAS", "DEVRAIT", "NE DEVRAIT PAS", "RECOMMANDE", "PEUT", et "FACULTATIF" en majuscules dans ce document sont à interpréter comme décrit dans le BCP 14, [RFC2119].

Les exigences formulées à l'impératif au titre des algorithmes (comme "supprimer tout caractère espace en tête" ou "retourner faux et interrompre ces étapes") sont à interpréter avec la même signification que les mots clés ("DOIT", "DEVRAIT", "PEUT", etc.) utilisés pour introduire l'algorithme.

Les exigences de conformité formulées comme des algorithmes ou des étapes spécifiques peuvent être mises en œuvre de n'importe quelle façon, pour autant qu'à la fin le résultat soit équivalent. En particulier, les algorithmes définis dans cette spécification sont destinés à être faciles à comprendre et ne sont pas destinés à être performants.

2.2 Notation de la syntaxe

La présente spécification utilise la notation du format Backus-Naur augmenté (ABNF) de la [RFC5234].

Le cœur des règles suivant est inclus par référence, comme défini dans la [RFC5234], Appendice B.1 : ALPHA (lettres), CR (retour chariot), CRLF (retour chariot saut à la ligne), CTL (caractères de contrôle), DIGIT (chiffres de 0 à 9), DQUOTE (guillemets), HEXDIG (chiffres hexadécimaux 0-9/A-F/a-f), LF (saut à la ligne), OCTET (toute séquence de données de huit bits), SP (espace), HTAB (tabulation horizontale), CHAR (tout caractère US-ASCII), VCHAR (tout

caractère US-ASCII visible), et WSP (espace blanche).

La règle OWS est utilisée lorsque zéro, un ou plusieurs octets d'espace blanche linéaire pourraient apparaître. Les OWS DEVRAIENT soit n'être pas produits, soit être produits comme un seul SP. Plusieurs octets d'OWS qui se produisent au sein d'un contenu de champ DEVRAIENT soit être remplacés par un seul caractère SP, soit transformés tout en octets SP (chaque octet autre que SP remplacé par SP) avant d'interpréter la valeur du champ ou de transmettre le message vers l'aval.

```
OWS = *( SP / HTAB / obs-fold )          ;  espace blanche "facultative"
obs-fold = CRLF ( SP / HTAB )            ; saut à la ligne obsolète
```

2.3. Terminologie

Les termes "agent d'utilisateur", "client", "serveur", "mandataire", et "serveur d'origine" ont la même signification que dans la spécification HTTP/1.1 ([RFC2616], paragraphe 1.3).

Un identifiant unique au monde est une valeur qui est différente de toutes les autres valeurs existant préalablement. Par exemple, une chaîne aléatoire suffisamment longue va probablement être un identifiant unique au monde. Si la valeur d'origine ne quitte jamais l'agent d'utilisateur, un compteur à accroissement monotone chez l'agent d'utilisateur peut aussi servir d'identifiant unique au monde.

3. Principes de la politique de même origine

De nombreux agents d'utilisateur entreprennent des actions au nom de parties distantes. Par exemple, Les agents d'utilisateur HTTP suivent les redirections, qui sont des instructions provenant des serveurs distants, et les agents d'utilisateur HTML exposent les interfaces du modèle d'objet documentaire (DOM, *Document Object Model*) riche aux scripts restitués des serveurs distants.

Sans aucun modèle de sécurité, les agents d'utilisateur pourraient entreprendre des actions au détriment de l'utilisateur ou d'autres parties. Au fil du temps, beaucoup des technologies en rapport avec la Toile ont convergé vers un modèle de sécurité commun, connu familièrement sous le nom de "politique de la même origine". Bien que ce modèle de sécurité ait largement évolué organiquement, la politique de la même origine peut être comprise dans les termes d'une poignée de concepts clés. Cette section présente ces concepts et donne des avis sur la façon d'utiliser ces concepts en toute sécurité.

3.1 Confiance

La politique de même origine spécifie la confiance par URI. Par exemple, les documents HTML désignent quel script faire fonctionner avec un URI : <script src="https://exemple.com/librairie.js"></script>

Quand un agent d'utilisateur traite cet élément, l'agent d'utilisateur va aller chercher le script à l'URI désigné et exécuter le script avec les privilèges du document. De cette façon, le document accorde tous les privilèges qu'il a à la ressource désignée par l'URI. Par nature, le document déclare qu'il fait confiance à l'intégrité des informations restituées de cet URI.

En plus d'importer des bibliothèques des URI, les agents d'utilisateur envoient aussi des informations aux parties distantes désignées par l'URI. Par exemple, considérons l'élément "form" HTML :

```
<form method="POST" action="https://exemple.com/login">
 ... <input type="password"> ...
</form>
```

Quand l'utilisateur entre son mot de passe et soumet le "form", l'agent d'utilisateur envoie le mot de passe au point d'extrémité réseau désigné par l'URI. De cette façon, le document exporte ses données secrètes à cet URI, déclarant par ce fait qu'il a confiance dans la confidentialité des informations envoyées à cet URI.

3.1.1 Pièges

Quand on conçoit de nouveaux protocoles qui utilisent la politique de la même origine, il faut s'assurer que les distinctions importantes de confiance sont visibles dans les URI. Par exemple, si à la fois des ressources protégées par la sécurité de la

couche transport (TLS, *Transport Layer Security*) et des ressources non protégées par TLS utilisent le schéma d'URI "http" (comme dans la [RFC2817]) un document va être incapable de spécifier qu'il souhaite restituer un script seulement sur TLS. En utilisant le schéma d'URI "https", les documents sont capables d'indiquer qu'ils souhaitent interagir avec les ressources qui sont protégées des attaquants actifs du réseau.

3.2 Origine

En principe, les agents d'utilisateur pourraient traiter chaque URI comme un domaine de protection séparé et exiger un consentement explicite pour la restitution de contenu à partir d'un URI pour interagir avec un autre URI. Malheureusement, cette conception n'est pas crédible pour les développeurs parce que les applications de la Toile consistent souvent en un certain nombre de ressources agissant de concert.

Les agents d'utilisateur groupent plutôt les URI ensemble dans des domaines de protection appelés "origines". En gros, deux URI font partie de la même origine (c'est-à-dire, représentent le même principal) si ils ont le même schéma, hôte, et accès. (Voir tous les détails à la Section 4.)

Question : Pourquoi ne pas juste utiliser l'hôte ?

Réponse : Inclure le schéma dans le couple "origine" est essentiel pour la sécurité. Si les agents d'utilisateur n'incluent pas le schéma, il n'y aura pas d'isolation entre http://exemple.com et https://exemple.com parce que les deux auront le même hôte. Cependant, sans cette isolation, un attaquant actif du réseau pourrait corrompre le contenu restitué de http://exemple.com et faire que ce contenu donne pour instruction à l'agent d'utilisateur de compromettre la confidentialité et l'intégrité du contenu restitué de https://exemple.com, outrepassant les protections offertes par TLS [RFC5246].

Question : Pourquoi utiliser le nom d'hôte pleinement qualifié au lieu de juste le domaine de "plus haut niveau" ?

Réponse : Bien que le DNS ait des délégations hiérarchiques, les relations de confiance entre les noms d'hôtes varient selon le déploiement. Par exemple, dans de nombreux établissement d'enseignement, les étudiants peuvent héberger du contenu à https://exemple.edu/~étudiant/,mais cela ne signifie pas qu'un document rédigé par un étudiant devrait faire partie de la même origine (c'est-à-dire, habiter le même domaine de protection) comme une application de la Toile pour gérer les diplômes hébergés à https://diplomes.exemple.edu/.

Le déploiement "exemple.edu" illustre que grouper des ressources par origine ne s'harmonise pas toujours parfaitement avec tous les scénarios de déploiement. Dans celui-ci, le site de la Toile de chaque étudiant habite la même origine, ce qui peut n'être pas souhaitable. Dans un certain sens, la granularité de l'origine est un artifice historique de la façon dont le modèle de sécurité a évolué.

3.2.1 Exemples

Toutes les ressources suivantes ont la même origine :
http://exemple.com/
http://exemple.com:80/
http://exemple.com/chemin/fichier

Chacun des URI a les mêmes composants schéma, hôte, et accès.

Chacune des ressources suivante a une origine différente des autres.
http://exemple.com/
http://exemple.com:8080/
http://www.exemple.com/
https://exemple.com:80/
https://exemple.com/
http://exemple.org/
http://ietf.org/

Dans chaque cas, au moins un des composants schéma, hôte, et accès va différer des autres de la liste.

3.3 Autorité

Bien que les agents d'utilisateur groupent les URI par origines, toutes les ressource dans une origine ne portent pas la même autorité (dans le sens de la sécurité du mot "autorité", pas au sens de la [RFC3986]). Par exemple, une image est un contenu passif et ne porte donc pas d'autorité, ce qui signifie que l'image n'a pas d'accès aux objets et ressources disponibles à son origine. À l'opposé, un document HTML porte la pleine autorité de son origine, et les scripts au sein du document (ou qui y sont importés) peuvent accéder à toute ressource dans son origine.

Les agents d'utilisateur déterminent combien d'autorité accorder à une ressource en examinant son type de support. Par exemple, des ressources avec un type de support image/png sont traitées comme des images, et des ressources avec un type de support text/html sont traitées comme des documents HTML.

Quand elles hébergent un contenu qui n'est pas de confiance (comme un contenu généré par un utilisateur) les applications de la Toile peuvent limiter l'autorité de ce contenu en restreignant son type de support. Par exemple, servir un contenu généré par un utilisateur comme image/png est moins risqué que de servir un contenu généré par un usager comme text/html. Bien sûr, de nombreuses applications de la Toile incorporent des contenus qui ne sont pas de confiance dans leurs documents HTML. Si ce n'est pas fait avec tout le soin nécessaire, ces applications risquent de divulguer l'autorité de leur origine au contenu non fiable, vulnérabilité couramment connue sous le nom de "écriture trans-site" *(cross-site scripting)*.

3.3.1 Pièges

Quand on conçoit de nouveaux éléments d'une plateforme de la Toile, il faut veiller à ne pas accorder d'autorité à des ressources sans considération du type de support. De nombreuses applications de la Toile servent des contenus non fiables avec des types de support restreints. Une nouvelle caractéristique de plateforme de la Toile qui accorde une autorité à ces éléments de contenu risque d'introduire des vulnérabilités dans les applications existantes. À la place, il faut préférer accorder l'autorité aux types de supports qui possèdent déjà la pleine autorité d'origine ou aux nouveaux types de supports conçus spécifiquement pour porter la nouvelle autorité.

Afin de rester compatible avec les serveurs qui fournissent des types de supports incorrects, certains agents d'utilisateur emploient un "reniflage de contenu" et traitent le contenu comme si il avait un type de support différent de celui fourni par le serveur. Si cela n'est pas fait avec une grande attention, le reniflage de contenu peut conduire à des faiblesses de la sécurité parce que les agents d'utilisateur peuvent accorder à des types de support de faible autorité, comme des images, les privilèges de types de support de haute autorité, comme ceux des documents HTML [SNIFF].

3.4 Politique

D'une façon générale, les agents d'utilisateur isolent les origines différentes et permettent une communication contrôlée entre les origines. Les détails de la façon dont les agents d'utilisateur fournissent l'isolation et la communication varient selon plusieurs facteurs.

3.4.1 Accès d'objet

La plupart des objets (aussi connus sous le nom d'interfaces de programmation d'application (API, *application programming interface*) exposés par l'agent d'utilisateur sont disponibles seulement à la même origine. Spécifiquement, le contenu restitué d'un URI peut accéder aux objets associés au contenu restitué d'un autre URI si, et seulement si, les deux URI appartiennent à la même origine, par exemple, ont le même schéma, hôte, et accès.

Il y a quelques exceptions à cette règle générale. Par exemple, certaines parties de l'interface de localisation de HTML sont disponibles à travers les origines (par exemple, pour permettre de naviguer sur d'autres contextes de navigation). Un autre exemple, l'interface postMessage de HTML est visible explicitement à travers les origines pour faciliter la communication trans origine. Exposer les objets aux origines étrangères est dangereux et devrait n'être fait que avec de grandes précautions parce que le faire expose ces objets à de potentielles attaques.

3.4.2 Accès réseau

L'accès aux ressources du réseau varie selon que les ressources sont dans la même origine que le contenu qui tente d'y accéder.

Généralement, lire des informations provenant d'une autre origine est interdit. cependant, il est permis à une origine d'utiliser certaines sortes de ressources restituées d'autres origines. Par exemple, il est permis à une origine d'exécuter un script, de rendre des images, et d'appliquer des feuilles de style provenant de toute origine. De même, une origine peut afficher du contenu provenant d'une autre origine, comme un document HTML dans une trame HTML. Les ressources du réseau peuvent aussi opter pour laisser d'autres origines lire leurs informations, par exemple, en utilisant le partage de ressources trans origines (CORS, *Cross-Origin Resource Sharing*) [CORS]. Dans ces cas, l'accès est normalement accordé sur la base de l'origine.

L'envoi d'informations à une autre origine est permis. Cependant, l'envoi d'informations sur le réseau dans des formats arbitraires est dangereux. Pour cette raison, les agents d'utilisateur restreignent les documents à l'envoi d'informations en utilisant des protocoles particuliers, comme dans une demande HTTP sans en-têtes personnalisés. Étendre l'ensemble des protocoles permis, par exemple, en ajoutant la prise en charge de WebSockets, doit être fait avec prudence pour éviter d'introduire des vulnérabilités [RFC6455].

3.4.3 Pièges

Chaque fois que les agents d'utilisateur permettent à une origine d'interagir avec les ressources d'une autre origine, ils suscitent des problèmes de sécurité. Par exemple, la capacité d'afficher des images provenant d'une autre origine laisse fuir leur hauteur et leur largeur. De même, la capacité d'envoyer des demandes sur le réseau à une autre origine suscite des vulnérabilités de contrefaçons de demandes trans-site [CSRF]. Cependant, les mises en œuvre d'agent d'utilisateur mettent souvent en balance ces risques avec les bénéfices de permettre l'interaction trans-origine. Par exemple, un agent d'utilisateur HTML qui a bloqué les demandes réseau trans-origine va empêcher ses utilisateurs de suivre les hyper liens, une caractéristique clé de la Toile.

Quand on ajoute de nouvelles fonctionnalités à la plateforme de la Toile, il peut être tentant d'accorder un privilège à une ressource mais de supprimer ce privilège à une autre ressource dans la même origine. Cependant, supprimer des privilèges de cette façon est inefficace parce que les ressources sans le privilège peuvent généralement obtenir le privilège de toutes façons parce que les agents d'utilisateur n'isolent pas les ressources au sein d'une origine. À la place, les privilèges devraient être accordés ou supprimées des origines comme un tout (plutôt que de discriminer entre les ressources individuelles au sein d'une origine) [BOFGO].

3.5 Conclusion

La politique de la même origine utilise les URI pour désigner des relations de confiance. Les URI sont groupés dans des origines, qui représentent des domaines de protection. Certaines ressources dans une origine (par exemple, un contenu actif) sont gratifiées de la pleine autorité de l'origine, tandis que d'autres ressources dans l'origine (par exemple, un contenu passif) ne sont pas gratifiées de l'autorité de l'origine. Le contenu qui porte l'autorité de son origine a l'accès aux ressources d'objets et de réseau au sein de sa propre origine. Ce contenu a aussi un accès limité aux ressources d'objets et de réseau des autres origines, mais ces privilèges trans-origine doivent être dessinés avec prudence pour éviter les faiblesses de sécurité.

4. Origine d'un URI

L'origine d'un URI est la valeur calculée par l'algorithme suivant :

1. Si l'URI n'utilise pas un élément hiérarchique comme autorité de dénomination (voir au paragraphe 3.2 de la [RFC3986]) ou si l'URI n'est pas un URI absolu, générer alors un identifiant unique au monde frais et retourner cette valeur

Note : Faire fonctionner plusieurs fois cet algorithme pour le même URI peut produire des valeurs différentes à chaque fois. Normalement, les agents d'utilisateur calculent l'origine de, par exemple, un document HTML, une fois, et utilisent cette origine pour les vérifications de sécurité ultérieures plutôt que de recalculer l'origine pour chaque vérification de sécurité.

2. Soit uri-scheme le composant de schéma de l'URI, converti en minuscules.

3. Si la mise en œuvre ne prend pas en charge le protocole donné par uri-scheme, générer alors un identifiant unique au monde frais et retourner cette valeur.

4. Si uri-schéma est "file", la mise en œuvre PEUT retourner une valeur qu'elle a définie.

Note : Historiquement, les agents d'utilisateur ont accordé au contenu provenant du schéma "file" une quantité énorme de privilèges. Cependant, accorder à tous les fichiers locaux des privilèges aussi larges peut conduire à des attaques d'escalade de privilèges. Certains agents d'utilisateur ont eu du succès en accordant aux fichiers locaux des privilèges fondés sur le répertoire, mais cette approche n'a pas été largement adoptée. D'autres agents d'utilisateur utilisent des identifiants uniques au monde pour chaque URL "file", ce qui est l'option la plus sûre.

5. Soit uri-host le composant hôte de l'URI, converti en minuscules (en utilisant le collationnement "i;ascii-casemap" défini dans la [RFC4790]).

Note: Le présent document suppose que l'agent d'utilisateur effectue le traitement d'internationalisation des noms de domaines dans les applications (IDNA, *Internationalizing Domain Names in Applications*) et la validation quand il construit l'URI. En particulier, le présent document suppose que "uri-host" va seulement contenir des étiquettes LDH parce que l'agent d'utilisateur aura déjà converti toutes les étiquettes non ASCII en leurs étiquettes A correspondantes (voir la [RFC5890]). Pour cette raison, les politiques de sécurité fondées sur l'origine sont sensibles à l'algorithme IDNA employé par l'agent d'utilisateur. Voir plus de détails au paragraphe 8.4.

6. Si il n'y a pas de composant accès de l'URI :
   1. Soit "uri-port" l'accès par défaut pour le protocole donné par uri-scheme.

   Autrement :
   2. Soit uri-port le composant d'accès de l'URI.

7. Retourner le triplet (uri-scheme, uri-host, uri-port).

5. Comparaison des origines

Deux origines sont "les mêmes" si, et seulement si elles sont identiques. En particulier :

- Si les deux origines sont des triplets schéma/hôte/accès, les deux origines sont les mêmes si, et seulement si, elles ont des schémas, hôtes, et accès identiques.

- Une origine qui est un identifiant unique au monde ne peut pas être la même qu'une origine qui est un triplet schéma/hôte/accès.

Deux URI sont de même origine si leur origine est la même.

Note : Un URI n'est pas nécessairement de même origine avec lui-même. Par exemple, un URI "data" [RFC2397] n'est pas de même origine que lui-même parce que les URI "data" n'utilisent pas une autorité de dénomination fondée sur un serveur et donc ont des identifiants uniques au monde comme origine.

6. Mise en série des origines

Cette section définit comment mettre en série une origine dans une chaîne Unicode [Unicode6] et dans une chaîne ASCII [RFC0020].

6.1 Mise en série Unicode d'une origine

La mise en série unicode d'une origine est la valeur retournée par l'algorithme suivant :

1. Si l'origine n'est pas un triplet schéma/hôte/accès, alors retourner la chaîne nulle (c'est-à-dire, la séquence de codets U+006E, U+0075, U+006C, U+006C) et interrompre ces étapes.

2. Autrement, le résultat est la partie schéma du triplet d'origine.

3. Ajouter la chaîne "://" au résultat.

4. Ajouter chaque composant de la partie hôte du triplet d'origine (converti comme suit) au résultat, séparé par des caractères point (".") U+002E :
   1. Si le composant est une étiquette A, utiliser à la place l'étiquette U correspondante (voir les [RFC5890] et [RFC5891]).
   2. Autrement, utiliser le composant tel quel.

5. Si la partie accès du triplet d'origine est différente de l'accès par défaut pour le protocole donné par la partie schéma du triplet d'origine :
   1. Ajouter un caractère deux-points (":") U+003A et l'accès donné, en base dix, au résultat.

6. Retourner le résultat.

6.2 Mise en série ASCII d'une origine

La mise en série ascii d'une origine est la valeur retournée par l'algorithme suivant :

1. Si l'origine n'est pas un triplet schéma/hôte/accès, retourner alors la chaîne nulle (c'est-à-dire, la séquence de codets U+006E, U+0075, U+006C, U+006C) et interrompre cette étape.

2. Autrement, soit la partie schéma le résultat du triplet d'origine.

3. Ajouter la chaîne "://" au résultat.

4. Ajouter la partie hôte au triplet d'origine pour le résultat.

5. Si la partie accès du triplet d'origine est différente de l'accès par défaut pour le protocole donné par la partie schéma du triplet d'origine :
   1. Ajouter un caractère deux-points (":") U+003A et l'accès donné, en base dix, pour avoir le résultat.

6. Retourner le résultat.

7. Champ d'en-tête Origine HTTP

Cette section définit les champ d'en-tête Origine HTTP.

7.1 Syntaxe

Le champ d'en-tête Origine a la syntaxe suivante :

```
origine                  = "Origin:" OWS liste-d'origine-ou-nulle OWS
liste-d'origine-ou-nulle = %x6E %x75 %x6C %x6C / liste-d'origine
liste-d'origine          = origine-mise-en-série *( SP origine-mise-en-série )
origine-mise-en-série    = schéma "://" hôte [ ":" accès ]          ; <schéma>, <hôte>, <accès> d'après la RFC 3986
```

7.2 Sémantique

Quand il est inclus dans une demande HTTP, le champ d'en-tête Origin indique la ou les origines qui ont "causé" la production de la demande par l'agent d'utilisateur, comme défini par l'API qui a déclenché la production de la demande par l'agent d'utilisateur.

Par exemple, considérons un agent d'utilisateur qui exécute des scripts au nom des origines. Si un de ces scripts cause la production par l'agent d'utilisateur d'une demande HTTP, l'agent d'utilisateur PEUT utiliser le champ d'en-tête Origine pour informer le serveur du contexte de sécurité dans lequel le script a été exécuté quand il a causé la production de la demande par l'agent d'utilisateur.

Dans certains cas, plusieurs origines contribuent à causer la production d'une demande HTTP par les agents d'utilisateur. Dans ces cas, l'agent d'utilisateur PEUT faire la liste de toutes les origines dans le champ d'en-tête Origine. Par exemple, si

la demande HTTP a été initialement produite par une origine mais ensuite redirigée par une autre origine, l'agent d'utilisateur PEUT informer le serveur que deux origines ont été impliquées dans la cause de la production de la demande par l'agent d'utilisateur.

7.3 Exigences pour l'agent d'utilisateur

L'agent d'utilisateur PEUT inclure un champ d'en-tête Origine dans toute demande HTTP.

L'agent d'utilisateur NE DOIT PAS inclure plus d'un champ d'en-tête Origine dans une demande HTTP.

Chaque fois qu'un agent d'utilisateur produit une demande HTTP à partir d'un contexte "sensible à la confidentialité", il DOIT envoyer la valeur "nulle" dans le champ d'en-tête Origine.

Note : Le présent document ne définit pas la notion de contexte sensible à la confidentialité. Les applications qui génèrent des demandes HTTP peuvent désigner des contextes comme sensibles à la confidentialité pour imposer des restrictions à la façon dont les agents d'utilisateur génèrent les champs d'en-tête Origine.

Quand il génère un champ d'en-tête Origine, l'agent d'utilisateur DOIT satisfaire aux exigences suivantes :
- Chacune des productions d'origine mise en série dans la grammaire DOIT être la mise en série ascii d'une origine.
- Deux productions consécutives d'origine mise en série dans la grammaire ne doivent pas être identiques. En particulier, si l'agent d'utilisateur devait générer deux origines consécutives mises en série, l'agent d'utilisateur NE DOIT PAS générer la seconde.

8. Considérations sur la sécurité

La politique de la même origine est une pierre angulaire de la sécurité pour de nombreux agents d'utilisateur, incluant les navigateurs de la Toile. Historiquement, certains agents d'utilisateur ont essayé d'autres modèles de sécurité, incluant le traçage de corruption et la prévention d'exfiltration, mais ces modèles se sont révélés difficiles à mettre en œuvre pour le moment (bien qu'il y ait eu un intérêt récent pour une renaissance de certaines de ces idées).

Évaluer la sécurité de la politique de la même origine est difficile parce que le concept d'origine lui-même joue un rôle si central dans le paysage de la sécurité. La notion d'origine elle-même est juste une unité d'isolation, imparfaite comme le sont la plupart des notions à taille unique. Ceci dit, il y a quelques faiblesses systémiques, qu'on discute ci-dessous.

8.1 Fiabilité du DNS

En pratique, la politique de la même origine s'appuie sur le système des noms de domaines (DNS, *Domain Name System*) pour la sécurité parce que de nombreux schémas d'URI d'utilisation courante, comme http, utilisent des autorités de dénomination fondées sur le DNS. Si le DNS est partiellement ou complètement compromis, la politique de même origine peut échouer à fournir les propriétés de sécurité demandées par les applications.

Certains schémas d'URI, comme https, sont plus résistants à la compromission du DNS parce que les agents d'utilisateur emploient d'autres mécanismes, comme des certificats, pour vérifier la source du contenu restitué de ces URI. D'autres schémas d'URI, comme le schéma d'URI chrome-extension (voir le paragraphe 4.3 de [CRX]) utilisent une autorité de dénomination fondée sur une clé publique et sont pleinement sûrs à l'égard de la compromission du DNS.

Le concept d'origine de la Toile isole les contenus restitués de schémas d'URI différents ; ceci est essentiel pour contenir les effets d'une compromission du DNS.

8.2 Unités d'isolation divergentes

Avec le temps, un certain nombre de technologies ont convergé sur le concept d'origine de la Toile comme une unité pratique d'isolation. Cependant, de nombreuses technologies en usage aujourd'hui, comme les mouchards *(cookies)* [RFC6265], datent d'avant le concept moderne d'origine de la Toile. Ces technologies ont souvent des unités d'isolation différentes, conduisant à des vulnérabilités.

Une solution de remplacement est de n'utiliser que le domaine "contrôlé par le registre" plutôt que le nom de domaine

pleinement qualifié comme unité d'isolation (par exemple, "exemple.com" au lieu de "www.exemple.com"). Cette pratique est problématique pour un certain nombre de raisons et N'EST PAS RECOMMANDÉE :

1. La notion de domaine "contrôlé par le registre" est une fonction des pratiques humaines entourant le DNS plutôt que une propriété du DNS lui-même. Par exemple, de nombreuses municipalités au Japon tiennent des registres publics assez poussés dans la hiérarchie du DNS. Ce sont des "listes de suffixes publics" largement utilisés, mais ces listes sont difficiles à tenir à jour et varient selon les mises en œuvre.

2. Cette pratique est incompatible les schémas d'URI qui n'utilisent pas une autorité de dénomination fondée sur le DNS. Par exemple, si un certain schéma d'URI utilise des clés publiques comme autorités de dénomination, la notion d'une clé publique "contrôlée par un registre" est un peu incohérente. Pire, certains schémas d'URI, comme nntp, utilisent des délégations séparées par des points dans le sens opposé de celui du DNS (par exemple, alt.usenet.kooks) et d'autres utilisent le DNS mais présentent les étiquettes à l'inverse de l'ordre usuel (par exemple, com.exemple.www).

Au mieux, utiliser des domaines "contrôlés par un registre" est spécifique du schéma d'URI et de la mise en œuvre. Au pire, les différences entre les schémas d'URI et les mises en œuvre peuvent conduire à des vulnérabilités.

8.3 Autorité ambiante

Quand ils utilisent la politique de même origine, les agents d'utilisateur accordent l'autorité au contenu sur la base de son URI plutôt que sur la base des objets que le contenu peut désigner. Ce désembroullage de désignation à partir de l'autorité est un exemple d'autorité ambiante et peut conduire à des vulnérabilités.

Considérons, par exemple, des scripts trans-site dans des documents HTML. Si un attaquant peut injecter un contenu de script dans un document HTML, ces scripts vont fonctionner sous l'autorité de l'origine du document, permettant peut-être l'accès du script à des informations sensibles, comme les enregistrements médicaux de l'usager. Si, cependant, l'autorité du script a été limitée aux objets que le script peut désigner, l'attaquant n'obtiendra pas d'avantage à injecter le script dans un document HTML hébergé par un tiers.

8.4 Dépendance à IDNA et migration

Les propriétés de sécurité de la politique de même origine peuvent dépendre de façon cruciale de détails des algorithmes IDNA employés par l'agent d'utilisateur. En particulier, un agent d'utilisateur pourrait transposer des noms de domaine internationaux (par exemple, ceux impliquant le caractère U+00DF) pour différentes représentations ASCII selon que l'agent d'utilisateur utilise IDNA2003 [RFC3490] ou IDNA2008 [RFC5890].

Migrer d'un algorithme IDNA à un autre pourrait redessiner un certain nombre de frontières en matière de sécurité, érigeant potentiellement de nouvelles frontières de sécurité, ou pire, en supprimant les frontières de sécurité entre deux entités qui ne sont pas de confiance l'une pour l'autre. Changer les frontières de sécurité est risqué parce que combiner deux entités qui ne sont pas de confiance l'une pour l'autre dans la même origine peut permettre à l'une d'attaquer l'autre.

9. Considérations relatives à l'IANA

Le registre permanent de champ d'en-tête de message (voir la [RFC3864]) a été mis à jour avec l'enregistrement suivant :

Nom de champ d'en-tête : Origin
Protocole applicable : http
Statut : standard
Auteur/contrôleur des changements : IETF
Document de spécification : la présente spécification (RFC6454)

10. Références

10.1 Références normatives

[RFC0020] V. Cerf, "Format ASCII pour les échanges sur les réseaux", octobre 1969. (STD80)

[RFC2119] S. Bradner, "Mots clés à utiliser dans les RFC pour indiquer les niveaux d'exigence", BCP 14, mars 1997. *(MàJ par* RFC8174)

[RFC2616] R. Fielding et autres, "Protocole de transfert hypertexte -- HTTP/1.1", juin 1999. *(D.S., MàJ par* 2817, 6585)

[RFC3864] G. Klyne, M. Nottingham, J. Mogul, "Procédures d'enregistrement pour les champs d'en-tête de message", septembre 2004. (BCP0090)

[RFC3986] T. Berners-Lee, R. Fielding et L. Masinter, "Identifiant de ressource uniforme (URI) : Syntaxe générique", STD 66, janvier 2005.

[RFC4790] C. Newman et autres, "Registre de collationnement des protocoles d'application de l'Internet", mars 2007. *(P.S.)*

[RFC5234] D. Crocker, éd., P. Overell, "BNF augmenté pour les spécifications de syntaxe : ABNF", janvier 2008. (STD0068)

[RFC5890] J. Klensin, "Noms de domaine internationalisés pour les applications (IDNA) : Définitions et cadre documentaire", août 2010. *(Remplace RFC*3490) *(P.S.)*

[RFC5891] J. Klensin, "Noms de domaine internationalisés pour les applications (IDNA) : Le protocole", août 2010. (*Remplace* RFC3490, RFC3491) (*MàJ* RFC3492) *(P.S.)*

[Unicode6] Unicode Consortium, "The Unicode Standard, Version 6.0.0", 2011, <http://www.unicode.org/versions/Unicode6.0.0/>.

10.2 Références pour information

[BOFGO] Jackson, C. et A. Barth, "Beware of Finer-Grained Origins", 2008, <http://w2spconf.com/2008/papers/s2p1.pdf>.

[CORS] van Kesteren, A., "Cross-Origin Resource Sharing", W3C Working Draft WD-cors-20100727, juillet 2010, <http://www.w3.org/TR/2010/WD-cors-20100727/>. Dernière version à <http://www.w3.org/TR/cors/>.

[CRX] Barth, A., Felt, A., Saxena, P., et A. Boodman, "Protecting Browsers from Extension Vulnerabilities", 2010, <http://www.isoc.org/isoc/conferences/ndss/10/pdf/04.pdf>.

[CSRF] Barth, A., Jackson, C., et J. Mitchell, "Robust Defenses for Cross-Site Request Forgery", 2008, <http://portal.acm.org/citation.cfm?id=1455770.1455782>.

[HTML] Hickson, I., "HTML5", W3C Working Draft WD-html5-20110525, mai 2011, <http://www.w3.org/TR/2011/WD-html5-20110525 >. Dernière version à <http://www.w3.org/TR/html5/>.

[RFC2397] L. Masinter, "Le schéma d'URL "data"", août 1998. *(P.S.)*

[RFC2817] R. Khare, S. Lawrence, "Mise à niveau de TLS au sein de HTTP/1.1", mai 2000. *(P.S.)*

[RFC3490] P. Faltstrom et autres, "Internationalisation des noms de domaine dans les applications (IDNA)", mars 2003. (*Remplacée par les* RFC5890 et 5891, *P.S.*)

[RFC5246] T. Dierks, E. Rescorla, "Version 1.2 du protocole de sécurité de la couche Transport (TLS)", août 2008. *(P.S.* ; *remplace* RFC3268, 4346, 4366 ; *MàJ* RFC4492 ; *rendue obsolète par la* RFC8446)

[RFC6265] A. Barth, "Mécanisme de gestion d'état HTTP", avril 2011. *(Remplace la* RFC2965) *(P.S.)*

[RFC6455] I. Fette, A. Melnikov, "Protocole WebSocket", décembre 2011. *(P.S. ; MàJ par* RFC7936, 8307, 8441)

[SNIFF] Barth, A. et I. Hickson, "Media Type Sniffing", Travail en cours, mai 2011.

Remerciements

Merci à Lucas Adamski, Stephen Farrell, Miguel A. Garcia, Tobias Gondrom, Ian Hickson, Anne van Kesteren, Jeff Hodges, Collin Jackson, Larry Masinter, Alexey Melnikov, Mark Nottingham, Julian Reschke, Peter Saint-Andre, Jonas Sicking, Sid Stamm, Daniel Veditz, et Chris Weber de leurs précieux retours sur le présent document.

Adresse de l'auteur

Adam Barth
Google, Inc.

mél : ietf@adambarth.com
URI : http://www.adambarth.com/