Retour d'expérience sur la conception d'une solution d'authentification renforcée

Dominique ALGLAVE
MENESR/SG/DNE/B1
61-65 rue Dutot
75015 Paris

Pascal COLOMBANI
MENESR/SG/DNE/B1
61-65 rue Dutot
75015 Paris

Résumé

Dans les systèmes d'information, la porte d'entrée est gardée grâce à l'authentification. Cette première brique de sécurité applicative est d'autant plus vulnérable qu'elle est assurée par une authentification unique (SSO), ce qui propage d'éventuelles faiblesses à l'ensemble du domaine fédéré.

L'administration centrale du MENESR développe une maquette d'une des solutions d'authentification renforcée étudiées (voir JRES 2013 - Etat de l'art de l'authentification renforcée) afin de valider le concept, son intégration, son niveau de sécurité, son ergonomie, l'enrôlement, l'organisation…

L'objectif visé est de renforcer l'authentification par rapport au mot de passe, c'est-à-dire rendre la solution plus résistante au phishing et au keylogger, sans pour autant être aussi contraignante à distribuer qu'une solution d'authentification forte ayant besoin d'un support physique.

Le ministère a fait le choix de développer en interne une maquette de grille dynamique car cette solution ne repose pas sur un dispositif physique, ne nécessite aucune installation sur le poste de travail de l'utilisateur et utilise un schéma spatial qui est plus difficilement transmissible qu'un mot de passe.

Le principe est que l'utilisateur connaît un schéma secret dans une grille générée dynamiquement et saisit les chiffres/lettres se trouvant dans les cases de ce schéma. Cela revient à créer un One Time Password (OTP). La force de l'authentification est renforcée grâce à un code « PIN » supplémentaire saisie avant ou après l'OTP.

La maquette expérimente le raccordement à des clients RADIUS, SAML V2 ou Open Id Connect pour une utilisation avec des finalités différentes (SSO Web, VPN, poste utilisateur…).

Mots-clefs

REX, Sécurité, Authentification renforcée, Grille dynamique, OTP, Fédération

1 Introduction

Dans les systèmes d'information intégrant toutes les briques technologiques à l'état de l'art, telles que la fédération d'identité et l'authentification unique (SSO), la faiblesse de l'authentification initiale se propage à l'ensemble du domaine fédéré. Ainsi, la force de l'authentification initiale se propage à l'ensemble du périmètre fédéré dans la mesure où le réseau fédéré est lui aussi sécurisé de manière adéquate. Le renforcement de l'authentification du périmètre du système d'information est donc un enjeu de premier ordre.

La validation de l'identité à l'accès est aujourd'hui assurée principalement par une authentification utilisant un mot de passe statique. Cette authentification dite faible est une source élevée de risques et est à l'origine de nombreux incidents d'usurpation d'identité. C'est pourtant le mode d'authentification plébiscité par les utilisateurs et celui pour lequel le plus d'informations et de préventions ont été réalisées.

Afin de protéger l'accès aux données ou aux sites web sensibles, le Ministère de l'Éducation Nationale utilise des moyens d'authentification dits forts tels que les clefs OTP ou les certificats sur carte à puce. La sécurité repose sur la possession d'un dispositif physique, augmenté de la connaissance d'un code PIN pour compléter ou déverrouiller celui-ci. La distribution de ces dispositifs demande la mise en place d'une logistique contraignante et coûteuse à laquelle peuvent se rajouter des ressources afin de gérer l'installation de lecteurs, de drivers et de logiciels.

L'authentification renforcée est la voie médiane qui permet d'obtenir un moyen d'authentification ayant certains critères de sécurité renforcés par rapport au niveau faible, tout en n'ayant pas toutes les contraintes d'un moyen fort. Ces solutions d'authentification offrent un compromis entre les différents critères de sécurité, de déploiement, d'adhérence, de coût, d'ergonomie, d'intégration et de pérennité.

Quelle solution renforcée serait la plus adaptée à une expérimentation dans le contexte du Ministère de l'Éducation Nationale ?

Une étude sur ces solutions renforcées [1] nous a permis de sélectionner la « grille dynamique ». Elle est apparue comme la plus facile à mettre en œuvre, car elle ne repose sur aucun composant à déployer du coté client. Le développement d'une maquette a été réalisé avec l'aide d'un stagiaire [2] dans le courant de l'année 2015.

2 Grille dynamique

2.1 Principe

Ce moyen d'authentification repose sur la validation d'une preuve que l'utilisateur connaît un secret sans que celui ci ne soit divulgué. Pour cela, on doit mettre en œuvre une grille de caractères (généralement des chiffres) générée aléatoirement et permettre à un utilisateur de choisir un schéma spatial secret qui représente une suite de positions sur la grille. En superposant le schéma spatial à la grille, on obtient une suite de caractères qui fait office de preuve. Le schéma spatial peut être vu comme un masque constitué d'une feuille de papier avec des trous qui permettrait de cacher la plupart des caractères afin de ne révéler que ceux constituant la preuve.

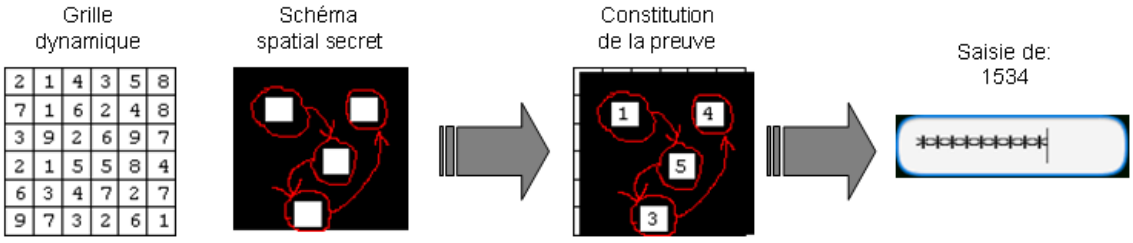

Figure 1 - Principe des grilles dynamiques

Même si visuellement elle ressemble grandement à la solution d'authentification proposée par les banques pour l'accès aux comptes en lignes, la grille dynamique n'est pas une solution de type clavier virtuel (l'utilisateur ne clique pas dessus, il lit les caractères correspondants).

La grille est changée régulièrement, ce qui lui confère son caractère dynamique. Si la grille est changée à chaque tentative de connexion, la preuve à fournir change à chaque fois et ressemble à l'utilisation d'un mot de passe à usage unique ou « *One Time Password* » (OTP). Dans ce cas, la capture de la preuve n'est d'aucun effet sur la sécurité comme pour l'OTP en général. Avec un jeu de caractères restreint (uniquement des chiffres) on obtient des répétitions du même symbole.

La capacité de la solution à résister à une énumération doit être envisagée à partir :

- du schéma spatial ; avec L et C le nombre de lignes et de colonnes et N le nombre de cases ; $Force = log_2((LxC) ! - ((LxC)-N) !)$
- de la preuve qui n'est valide qu'une minute ; avec S nombre de symboles et N nombre de caractères ; $Force = log_2(S \hat{} N)$

La force d'une grille peut être comparée à celle d'un mot de passe usuel. Lorsqu'il est composé de 8 caractères (minuscules, majuscules, chiffres et caractères spéciaux) sa force est de 49 [3]. Une grille de 9x9 composée uniquement de chiffres pour un schéma de 8 cases a une force de 56 et la preuve a une force de 26.

Aucun support matériel ou logiciel n'est à déployer et le poste sur lequel on utilise la solution peut être partagé. La saisie de la preuve s'effectue dans un champ standard de type mot de passe qui masque la saisie ce qui permet son utilisation en salle de classe sur un tableau numérique.

2.2 Défauts et vulnérabilités

2.2.1 Ergonomie

La solution de grille dynamique nécessite un enchaînement d'actions par l'utilisateur. L'authentification se déroule souvent en plusieurs étapes car elle est utilisée comme un deuxième facteur. La cinématique est généralement la suivante :

- identification de l'utilisateur grâce au login ou authentification avec login/mot de passe ;
- affichage d'une grille générée en fonction de l'utilisateur et demande de saisie de la preuve ;
- vérification de la preuve avec la grille générée pour cet utilisateur et renvoi vers le service.

De plus, un utilisateur peut rencontrer des difficultés à retenir un schéma spatial ayant beaucoup de positions et également à se repérer dans une grande grille. La solution n'est pas adaptée aux personnes en situation de handicap visuel.

2.2.2 Force du secret

Le secret peut être découvert après plusieurs observations en comparant les positions de plusieurs preuves sur les grilles correspondantes. En effet, un attaquant capable de récupérer la grille et la saisie clavier pourrait déduire au bout de quelques itérations le schéma spatial secret en comparant les différentes positions possibles sur la grille.

La découverte du schéma spatial peut être retardée en augmentant la redondance des symboles sur la grille. Pour cela, on peut :

- augmenter la taille de la grille, mais on rajoute de la difficulté pour l'utilisateur à se repérer dans la grille ;
- diminuer le nombre de symboles utilisés [3], mais cela réduit d'autant la capacité de la solution à résister à une énumération de toutes les preuves possibles et donc à une attaque en force brute.

2.2.3 Protection des secrets

Le stockage des schémas spatiaux ne peut être effectué sous forme d'un condensat (condensat MD5, SHA1 ou SHA256) comme pour les mots de passe. Or, le résultat de l'application d'une fonction de hâchage n'étant pas bijectif, il n'est pas possible de déduire le secret à partir de son condensat [4].

Par conséquent, en n'utilisant pas cette protection, le risque en cas de vol de la base d'un référentiel de comptes est l'exploitation directe des secrets par un attaquant.

3 Maquette

3.1 Objectifs

Le développement est une étude de faisabilité qui vient en continuité de l'étude sur les solutions d'authentifications renforcées. L'objectif principal est l'évaluation de la charge d'intégration d'une nouvelle solution d'authentification dans nos infrastructures. Il a été réalisé en plusieurs étapes afin de renforcer les fonctionnalités et la sécurité par lot suivant un processus agile. Chaque étape a permis d'avoir un résultat fonctionnel et déployable avec la documentation associée pour les interfaces. La durée de l'étude étant fixée à 3 mois, l'objectif était de développer les fonctionnalités essentielles pendant cette durée. Des comités hebdomadaires, réunissant un développeur stagiaire, le RSSI, le FSSI, le responsable du pôle « identités » et un expert du bureau des infrastructures, permettaient de fixer ces priorités.

La liste des objectifs initiaux fixés pour la réalisation de la maquette :

- réaliser une fonction de génération d'une grille dynamique paramétrable et sa représentation graphique la plus lisible et ergonomique ;
- réaliser une fonction de vérification de la preuve en fonction du schéma spatial de l'utilisateur ;
- trouver une solution pour l'enrôlement permettant à un utilisateur de créer son schéma spatial initial ;
- réaliser une fonction permettant de changer son schéma spatial de manière sécurisée ;
- optimiser la génération des grilles dynamiques pour supporter un fort volume de connexions simultanées ;
- réaliser des fonctions pour les administrateurs de la solution (configuration, création de compte, activation, blocage…) ;
- réaliser le raccordement avec la solution de contrôle d'accès utilisé pour les applications nationales du Ministère ;
- réaliser une interface d'authentification utilisant le protocole RADIUS ;
- sécuriser la partie client pour réduire les risques d'attaque par *keylogger* et capture d'écran, par injection SQL et XSS, ou par force brute ;
- sécuriser la partie serveur pour réduire les risques de vol de données.

3.2 Description

La maquette est composée de 4 applications développées en JAVA fonctionnant dans un serveur d'application compatible Java EE. Les données sont stockées dans une base de données ayant un driver compatible JDBC et respectant SQL:2008 (notamment pour LIMIT et OFFSET).

Un serveur TOMCAT a été choisi, mais les applications sont compatibles avec d'autres serveurs Java EE. Pour la maquette, nous avons déployé les 4 applications sur un seul serveur mais l'infrastructure cible comportera autant de serveurs que d'applications afin de permettre un meilleur filtrage des flux réseaux qui renforcera la sécurité.

La maquette embarque une base de données SQLite Java afin d'alléger l'infrastructure. La compatibilité n'étant assurée que pour SQL 92, le code détecte la non-compatibilité avec la fonction LIMIT et émule celle-ci. Cette méthode n'est pas recommandée en production car elle est peu performante; un serveur de base de données installé sur une machine dédiée permettra de traiter un grand volume d'utilisateurs.

Un serveur FreeRADIUS raccordé au moteur de grille permet de fournir une interface RADIUS à la solution.

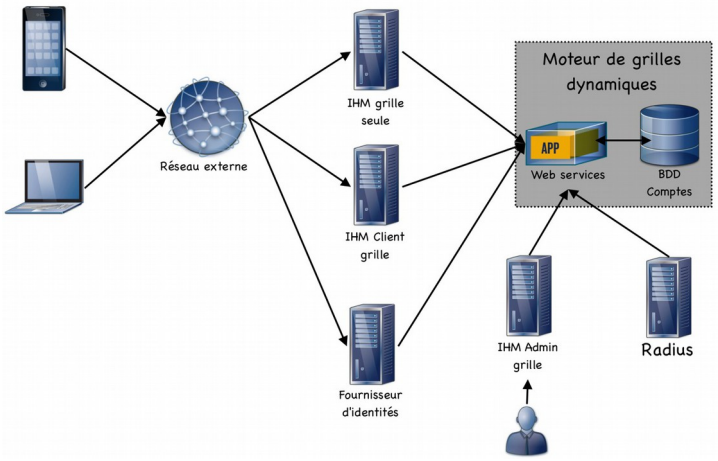

Figure 2 - Architecture cible de la solution

3.2.1 Coté client

L'interface web est développée à partir de la bibliothèque Google Web Toolkit (GWT).

Trois applications ont été développées :

- l'application à destination des utilisateurs permettant l'authentification, l'enrôlement et changement de schéma spatial. Elle s'appuie sur les web-services du moteur de grille pour fournir des services à l'utilisateur ;

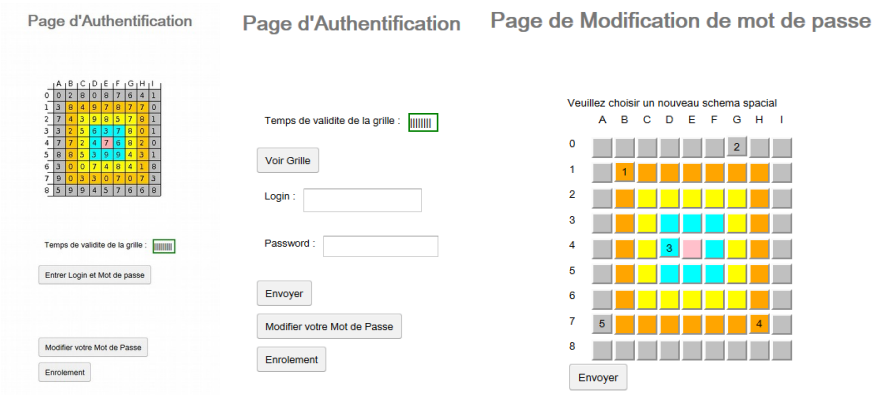

Figure 3 - interface de l'application d'authentification / enrôlement

- l'application de présentation de la grille. Elle permet de présenter la grille à un utilisateur afin qu'il saisisse la preuve dans une interface standard demandant seulement un login et un mot de passe, la preuve servant de mot de passe. Un exemple d'utilisation serait un service VPN configuré pour utiliser une authentification standard que l'on renforce par une grille. L'application s'appuie sur les web-services du moteur de grille pour récupérer la représentation graphique de la grille ;

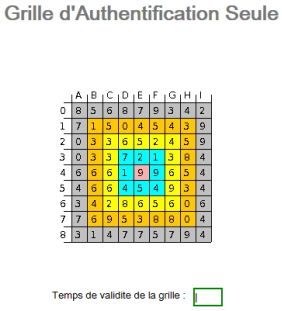

Figure 4 - IHM de présentation de la grille seule

- l'application d'administration qui permet de configurer les paramètres de grille, la configuration de la solution et la gestion des comptes. Elle s'appuie sur les web-services d'administration pour fournir des services à un administrateur.

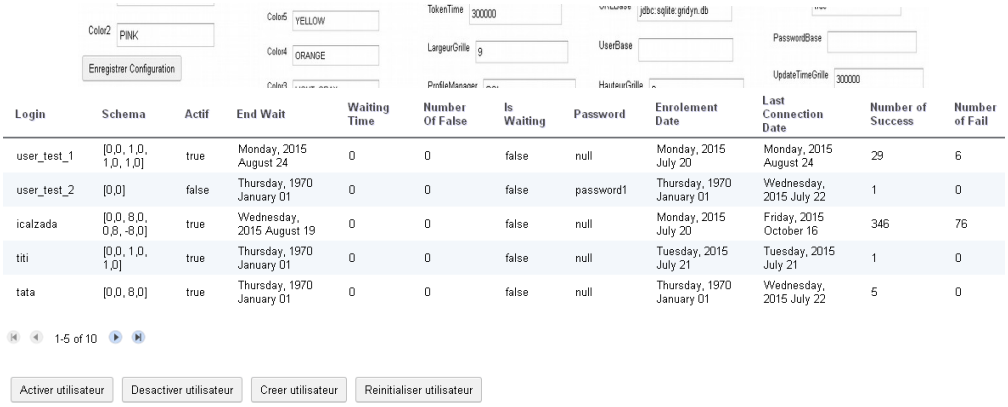

Figure 5 - interface d'administration de la solution

Les applications dépendent de l'application serveur pour fonctionner, mais ne sont qu'un exemple d'implémentation. Il faudra soit les adapter, soit développer une nouvelle version intégrable dans le fournisseur d'identités ou le portail d'une académie ou d'un établissement.

Les interfaces d'administration ont fait partie des derniers développements et n'ont pas bénéficié d'une passe d'amélioration de la sécurité et de l'ergonomie. Elles fournissent juste les services essentiels pour le fonctionnement de la partie administration de la maquette.

3.2.2 Coté serveur

L'application serveur fournit un ensemble de web-services qui offrent les fonctionnalités de base pour la solution de grille dynamique. Ils ne doivent pas être directement accessibles, mais être utilisés afin de construire une solution personnalisée de grille dynamique.

Les web-services sont appelés principalement sous forme de requêtes POST, sauf pour ceux fournissant l'image de la grille ou sa durée de validité qui utilisent des requêtes GET. Les paramètres de requêtes et de réponses peuvent être dans un format XML ou JSON.

Les web-services dédiés à l'interface utilisateur sont les suivants :

- récupération de l'image de la grille courante ;
- récupération des informations sur la grille (taille et couleurs) ;
- vérification de l'authentification à partir du login et de la preuve ;
- vérification du secret pour valider l'enrôlement et enregistrer un nouveau schéma spatial pour l'utilisateur ;
- modification du schéma spatial d'un utilisateur avec authentification.

Les web-services de raccordement sont une adaptation du web-service d'authentification permettant de faire une façade qui s'adapte au type d'appel web-service prévu par le partenaire raccordé.

Les web-services d'administration sont les suivants :

- authentification d'un administrateur (par grille) et création d'une session ;
- récupération des informations de configuration ;
- modification des informations de configuration ;
- création d'un utilisateur ;
- désactivation/activation d'un utilisateur pour bloquer l'authentification de cet utilisateur ;
- réinitialisation d'un utilisateur pour forcer un nouvel enrôlement.

3.3 Résultats

La maquette a permis de résoudre les défauts et vulnérabilités présentés dans les paragraphes précédents.

3.3.1 Ergonomie

La taille des grilles est paramétrable dans le moteur de grilles et permet donc de l'adapter en fonction du besoin pour un fournisseur d'identités. Il est recommandé de choisir ce paramétrage avant la création du premier schéma spatial, mais une extension de la largeur ou/et de la hauteur est possible sans perturber l'utilisation des schémas spatiaux existants dans la base. Un schéma spatial étant une suite de déplacement dans la grille, l'agrandissement de celle-ci ne pose pas de problème de calcul de la preuve, mais crée une changement visuel pour l'utilisateur.

Une grille de taille de 9x9 semble un bon compromis entre sécurité et ergonomie pour les quelques personnes qui ont testé la maquette. Avec cette taille de grille, la force d'un schéma spatial de 8 cases est équivalente à celle d'un mot de passe de 9 caractères.

Afin d'améliorer la lisibilité de la grille, une numérotation des lignes et des colonnes a été ajoutée à la représentation graphique. Une coloration permet également de repérer plus facilement les lignes et les colonnes et facilite la recherche des cellules du schéma spatial par un utilisateur.

Les couleurs sont paramétrables afin de définir ce qui est le plus adapté à la taille de la grille. Il est possible de définir un modèle de couleurs adapté aux différents types de daltonismes en alternant la couleur blanche avec un gris par exemple.

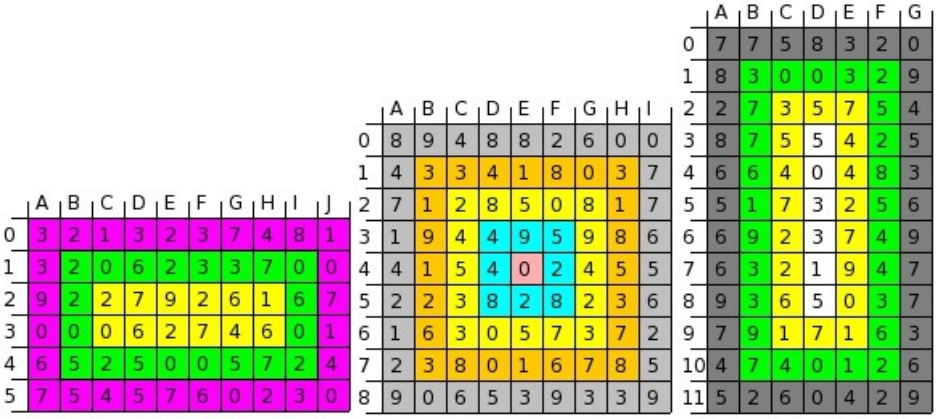

Figure 6 - Grille dynamique paramétrable

3.3.2 Optimisation de la génération de grille dynamique

La génération d'une grille est réalisée en plusieurs étapes : initialisation d'une fonction pseudo-aléatoire, génération d'un tableau de symboles en mémoire, création d'une image au format jpeg et sauvegarde du fichier jpeg en cache mémoire.

Afin de supporter un grand nombre de connexions simultanées, la génération de la grille n'est pas réalisée pour chaque connexion ou pour chaque utilisateur.

La solution mise en œuvre génère trois grilles qui seront utilisées pour toutes les connexions. Toutes les minutes, une nouvelle grille est générée et vient s'insérer et décaler l'ordre des trois grilles ; la grille la plus ancienne est supprimée. Ces trois grilles représentent l'ancienne grille valide, la grille actuellement valide et la prochaine grille qui sera utilisée.

Cette solution implique d'informer l'utilisateur du temps restant pour utiliser la grille, l'objectif étant d'éviter que l'utilisateur ne constitue une preuve à partir des générations de deux grilles successives.

La vérification de la preuve est effectuée sur la grille courante, mais également sur la grille précédente afin de ne pas pénaliser un utilisateur qui validerait sa saisie au moment du changement de grille. Ce fonctionnement rend une grille valide pendant deux fois plus de temps et laisse donc deux fois plus de temps à un attaquant pour tenter une attaque de type force brute.

La charge processeur induite par la génération des grilles ne dépend plus du nombre d'utilisateurs qui se connectent mais est optimisée pour ne survenir qu'un fois par minute.

3.3.3 Réduction du risque d'attaque

Afin de renforcer la sécurité de l'application, on utilise un outil d'audit de code automatique qui permet de détecter les vulnérabilités dans le code de type injection, XSS… [5]. Les vulnérabilités détectées ont été corrigées à chaque itération du développement.

La protection contre une attaque en force brute est réalisée en ralentissant l'attaquant à chaque tentative qui échoue. Le compte est bloqué pendant un temps d'attente 2 fois plus long à chaque échec. Ainsi, une attaque de type force brute ne peut énumérer toutes les combinaisons possibles de preuve dans le temps de vie d'une grille, ce qui ralentit très fortement la découverte du schéma spatial. Par contre, la solution devient plus sensible à une attaque en déni de service.

La solution de grille est résistante aux attaques par observation standard ; cependant, certains logiciels espions ne récoltent pas uniquement les saisies clavier mais effectuent également une capture d'écran. Une seule observation n'est pas suffisante pour en déduire le schéma spatial. Pour gêner ces logiciels :

- la grille est présentée sous forme d'une image ; il faut donc mettre en place de la reconnaissance de caractères pour automatiser la tâche de découverte ;
- lors de la saisie du login et de la preuve, la grille n'est pas affichée à l'écran (voir 3.2.1). Cela peut limiter l'impact sur les logiciels qui déclenchent la capture à chaque frappe clavier ;
- les symboles utilisés sont réduits aux chiffres et la taille de la grille est fixée à 9x9 ; chaque symbole est répété en moyenne 8 fois dans une grille ce qui augmente la résistance à l'observation.

3.3.4 Raccordements

La solution devait rester simple, centrée sur l'authentification et s'appuyer sur un référentiel de comptes dédié plutôt que sur un référentiel d'identités. Le développement d'une solution implémentant un raccordement en SAML v2 n'était pas pertinent car cela demanderait plus d'informations provenant du profil de l'utilisateur pour constituer un vecteur d'identité.

Le raccordement avec le produit Access Manager de RSA a été réalisé en développant un plug-in RSA interrogeant le web-service d'authentification de la grille dynamique. Le plug-in était très simple à réaliser et a demandé très peu de code.

L'utilisation d'un serveur FreeRADIUS 3.0.9 a simplifié l'implémentation du protocole Radius. Le serveur a été configuré pour rediriger les demandes d'authentification vers un web-service développé spécifiquement pour respecter le protocole d'appel « REST » de FreeRadius. La solution a été testée avec un client RADIUS en ligne de commande et en utilisant l'interface affichant seulement la grille dynamique.

3.3.5 Vol de données sur les serveurs

Différentes solutions de chiffrement et de coffres ont été envisagées, mais aucune n'a pu être mise en œuvre par manque de temps. Une solution de coffre sur une infrastructure virtuelle sera testée à l'avenir.

L'ajout d'un code PIN permettrait de renforcer les informations à retrouver sur le serveur. Le schéma spatial est en clair dans la base, mais le code PIN pourrait être hâché avec un sel et un nombre d'itérations. Ainsi, la seule possession du schéma spatial ne suffira plus pour pouvoir s'authentifier.

Afin de rester sur le même fonctionnement avec un couple de champs standards login et mot de passe, le code PIN est saisi à la suite de la preuve. Pour aller plus loin, en se raccordant au service LDAP académique ou de l'établissement, le mot de passe utilisateur pourrait être utilisé en remplacement du code PIN.

4 Conclusion

La solution retenue dans cette étude est basée sur le développement d'un maquette d'authentification renforcée au moyen de grilles dynamiques. Elle présente les avantages d'être plus résistante qu'une solution utilisant des mots de passe, de ne pas utiliser de dispositifs physiques et de logiciels du côté client. Cependant, l'appropriation par les utilisateurs de ce type de solution reste un frein dans le sens où les mécanismes sous-jacents sont encore relativement complexes à appréhender.

Le développement de la maquette nous a permis de tester différentes méthodes de raccordement qui pourraient être utilisés pour d'autres solutions d'authentification libre, « maison » ou commerciale. Cela étant dit, il serait intéressant de poursuivre le développement avec les technologies utilisées dans les portails de l'enseignement supérieur.

Annexe

Extrait de la documentation :

Authentification

URL :

```
http://votreadresse/GriDynWS/gridyn/ws/authenticate? +  paramètre facultatif
```

Le paramètre facultatif est :

```
&format=<JSON ou XML> //String
```

Requête :

Les requêtes peuvent se faire au format JSON ou XML. Si le paramètre facultatif n'est pas positionné, le format est XML.

La requête contient les éléments suivants :

action : doit être égal à « authentificate »

login : le login de l'utilisateur

password : le password de l'utilisateur

Format de la requête en XML :

```
<Request action="authenticate">
   <login></login>
   <password></password>
</Request>
```

Format de la requête en JSON :

```
{"action":"authenticate","login":"","password":""}
```

Réponse :

La réponse de l'API est au format XML par défaut ou au format indiqué en paramètre de la requête.

La réponse contient les éléments suivants :

return : le résultat de l'authentification (« OK » ou « NOK »)

errorcode : le code de l'erreur (uniquement si return= « NOK »)

locktime : le temps restant où le compte de l'utilisateur est bloqué (uniquement si errorcode=2)

Si l'un des éléments obligatoires de la requête est oublié, le return sera « NOK ».

Si le login ou le mot de passe est erroné ou si l'utilisateur n'a pas été trouvé dans la base, le return sera « NOK ».

Si le login est valide mais que le compte n'est pas actif, le return sera « NOK » et le errorcode sera «1».

Si le compte est bloqué pendant un certain temps car le nombre de tentatives de connexion échouées est supérieur à 3, le return sera « NOK », le errorcode sera «2 » et le locktime sera « jours – HH :MM :SS ». Le locktime représente le nombre de jours, heures, minutes et secondes restant à attendre afin que l'utilisateur puisse de nouveau tenter de s'authentifier.

Format de la réponse en XML :

```
<Response return="OK|NOK">
   <errorcode></errorcode>
   <locktime></locktime>
</Response>
```

Format de la réponse en JSON :

```
{"return":"","errorcode":"","locktime":""}
```

Exemple de Code : module permettant l'intégration avec le fournisseur d'identités RSA Access Manager

```
package org.men.GriDyn;

import java.io.BufferedWriter;
import java.io.InputStreamReader;
import java.io.OutputStreamWriter;
import java.net.HttpURLConnection;
import java.net.URL;
import java.util.Map;
import java.util.ResourceBundle;

import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;

import org.w3c.dom.Document;
import org.w3c.dom.Node;
import org.xml.sax.InputSource;

import sirrus.runtime.ResultConstants;
import sirrus.runtime.customauth.Authenticator;

public class GriDynAuth implements Authenticator
{
    public static String S_URL_LABEL = "GriDynSrvUrl";
    private String m_UrlGriDynServer;

    public GriDynAuth()
    {

        // read resourcebundle for URL
        ResourceBundle rb = ResourceBundle.getBundle("properties.conf");
        m_UrlGriDynServer=rb.getString(S_URL_LABEL);
        if(m_UrlGriDynServer==null)
        {
            //erreur: sysout("expliquerqu'il faut le conf.properties")
            System.out.println("ERREUR : Le fichier conf.properties n'a pas pu etre trouve ou le champs GriDynSrvUrl
est absent !");
        }
    }

    @Override
    public boolean authenticate(Map arg0, Map arg1)
    {
        String login = (String)arg0.get("SC_USER_ID");
        String pwd = (String)arg0.get("CREDENTIALS");
        if(authGrid(login, pwd))
        {
            arg1.put(ResultConstants.AUTHENTICATION_RESULT, ResultConstants.VALID_USER);
            return true;
        }
        else
        {
            arg1.put(ResultConstants.AUTHENTICATION_RESULT, ResultConstants.SECURID_AUTH_FAILED);
            return false;
        }
    }

    private boolean authGrid(String p_login, String p_pwd)
    {
        //appel api authentificate
        try {
            String XMLData = "<Request
action=\"authenticate\"><login>"+p_login+"</login><password>"+p_pwd+"</password><schema></schema></Request>";


            URL url = new URL(m_UrlGriDynServer);
            HttpURLConnection httpConn = (HttpURLConnection)url.openConnection();
            httpConn.setRequestMethod("POST");
            httpConn.setRequestProperty("Content-Type", "application/xml; charset=utf-8");
            httpConn.setRequestProperty("Content-Length", Integer.toString(XMLData.length()));
            httpConn.setDoOutput(true);

            BufferedWriter writer = new BufferedWriter(new OutputStreamWriter(httpConn.getOutputStream(),"UTF-8"));
            writer.write(XMLData, 0, XMLData.length());
            writer.flush();
            writer.close();

            DocumentBuilder db;
            try
            {
                db = DocumentBuilderFactory.newInstance().newDocumentBuilder();
                InputSource is = new InputSource(new InputStreamReader(httpConn.getInputStream(), "UTF-8"));

                Document doc = db.parse(is);
                System.out.println(doc.getFirstChild().getNodeName());
                Node l_res = doc.getFirstChild();

                if(l_res.getAttributes().item(0).getNodeValue().equalsIgnoreCase("OK"))
                {
                    return true;
```

```
                }
                else
                {
                    return false;
                }
            } catch (Exception e)
            {
                return false;
            }
        } catch (Exception e)
        {
            return false;
        }
    }

    @Override
    public String getAuthenticationType()
    {
        return "GRIDYN";
    }

}
```

Bibliographie

[1] Etat de l'art de l'authentification renforcée, JRES 2013 - https://2013.jres.org/archives/34/index.htm

[2] Rapport de stage d'Inaki Calzada; disponible sur demande auprès des auteurs.

[3] Calculer la « force » d'un mot de passe - http://www.ssi.gouv.fr/administration/precautions-elementaires/calculer-la-force-dun-mot-de-passe/

[4] A.J. Menezes, S.A. Vanstone, and P.C. Van Oorschot. Handbook of Applied Cryptography. CRC Press, Inc., Boca Raton, FL, USA, 1996.

[5] OWASP Top Ten Project - https://www.owasp.org