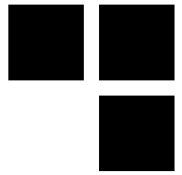
SYNACKTIV
DIGITAL SECURITY

Comment qu'est-ce qu'on flag ?

Introduction aux CTF (Capture The Flag)

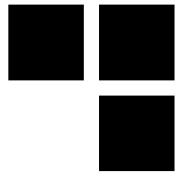
Présenté 23/04/2018
Pour 42
Par Corentin BAYET, Lucas ARRIVE

Comment qu’est-ce qu’on flag ?



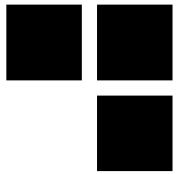
- **C’est quoi un CTF ?**
- **Les différents types de CTF**
  - Jeopardy, Attaque-défense, …
  - Les différentes épreuves
- **Commencer les CTF**
  - Les trucs à connaître
  - Rejoindre une équipe
- **Organiser un CTF**

What The CTF ?



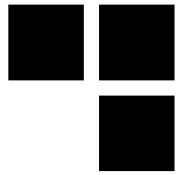
- **Chaque CTF est unique en soi**
- **Mais partagent tous un modèle commun**
- **Pas de règle fixe**

What The CTF ?



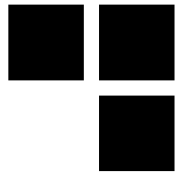
- **Modèle basique**
  - En équipe
  - Challenges de sécurité qui donnent des « flags »
  - Flags qui rapportent des points
  - L‘équipe ayant le plus de points gagne
- **Généralement entre 24 et 48h**
  - On échange du sommeil contre des flags
  - Moins long pour les CTF sur place
- **Des prix pour les premiers**
  - Des places pour la conf, qualification pour une finale

Les types de CTF - Jeopardy



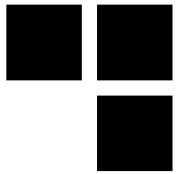
- **Challenges de différents types**
  - Reverse, Pwn, Web, Crypto, Network, Stegano, Forensic, Dev, Misc, ...
- **Le plus commun**
  - Format classique pour des qualifications
  - Facile à mettre en place en ligne
- **Différents systèmes de score :**
  - Statique
  - Dynamique en fonction du nombre de validations
  - First blood

Web



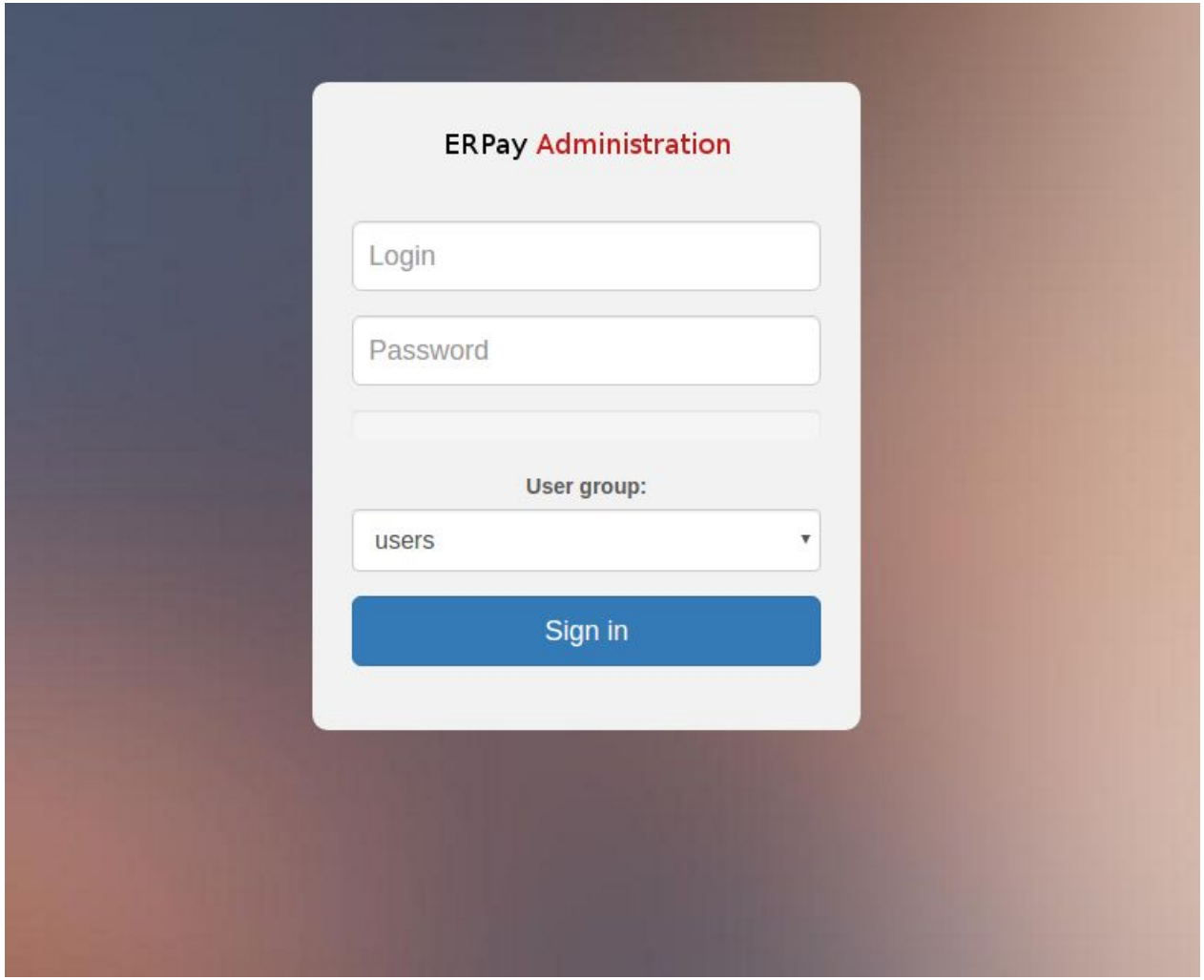
- **Une application web est mise à disposition**
- **Une ou plusieurs vulnérabilité(s) « web »**
  - On est en souvent « blackbox »
- **Le but de l'épreuve est variable :**
  - Devenir administrateur de l'application
  - Obtenir de l'exécution de code
  - Lire un fichier
  - Lire une base de données
  - ...

Web

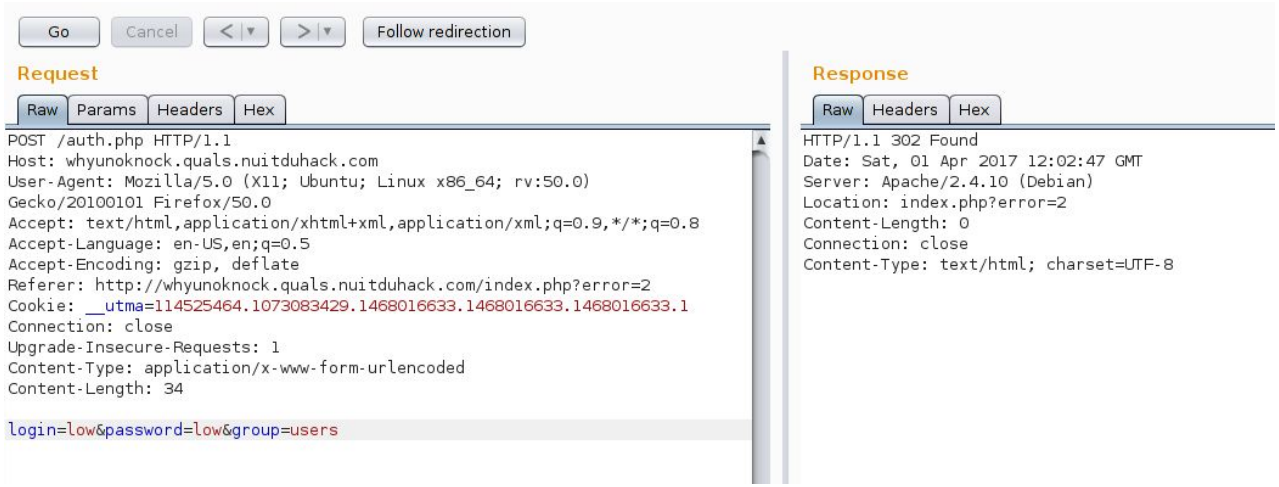


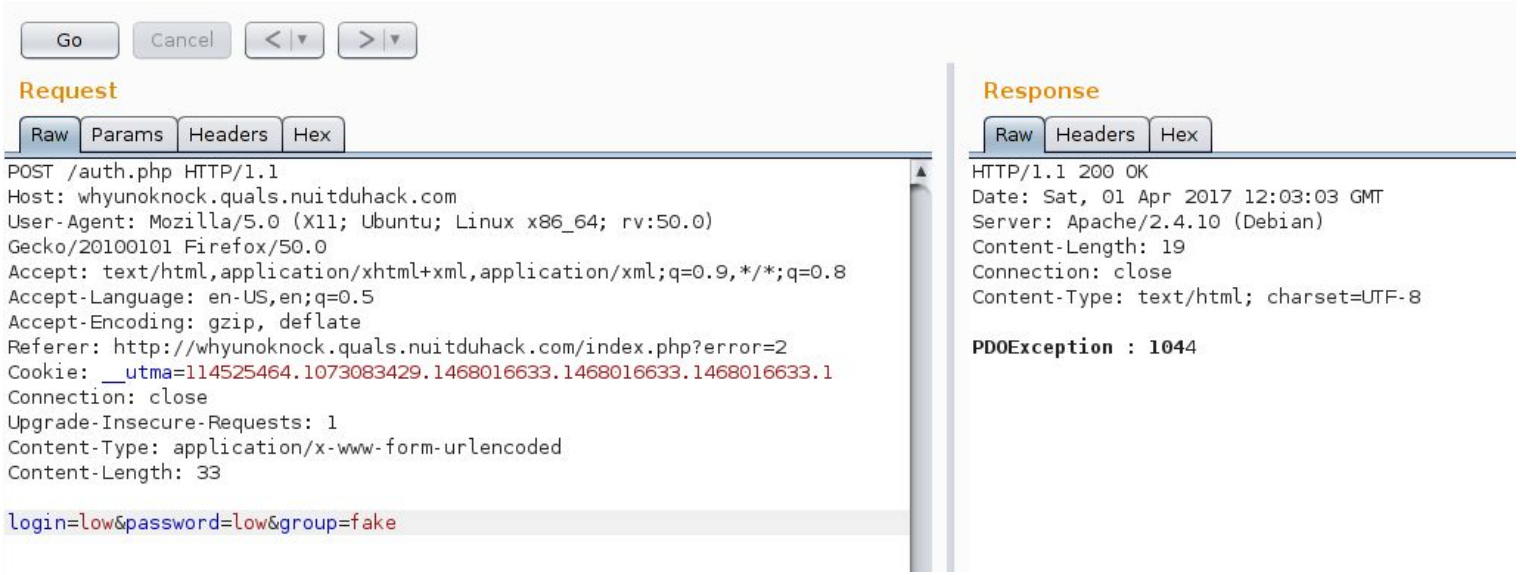
- **Nuit du Hack CTF Quals 2017**
- **WhyUNOKnock**
  - Injection dans l'argument « Data Source Name » de la classe PDO dans php
  - Permet de contrôler l'IP du serveur MySQL sur lequel se connecte le serveur web
  - En créant une fausse BDD MySQL on est capable de créer un utilisateur sur l'application et de s'authentifier

Web - WhyUNOKnock

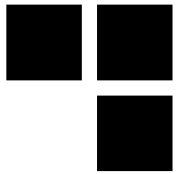


Web - WhyUNOKnock





Web - WhyUNOKnock



Description

```
public PDO::__construct ( string $dsn [, string $username [, string $password [, array $options ]]] )
```

Crée un objet PDO qui représente une connexion à la base.

Liste de paramètres

dsn
: Le *Data Source Name*, ou DSN, qui contient les informations requises pour se connecter à la base.

Description

Le Data Source Name (DSN) de PDO_MYSQL est composé des éléments suivants :

Préfixe DSN
: Le préfixe DSN est **`mysql:`**.

host
: L'hôte sur lequel le serveur de base de données se situe.

port
: Le numéro de port où le serveur de base de données est en train d'écouter.

dbname
: Le nom de la base de données.

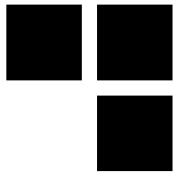
Exemple #1 Crée une instance PDO via une invocation de pilote

```php
<?php
/* Connexion à une base ODBC avec l'invocation de pilote */
$dsn = 'mysql:dbname=testdb;host=127.0.0.1';
$user = 'dbuser';
$password = 'dbpass';

try {
    $dbh = new PDO($dsn, $user, $password);
} catch (PDOException $e) {
    echo 'Connexion échouée : ' . $e->getMessage();
}

?>
```

Web - WhyUNOKnock

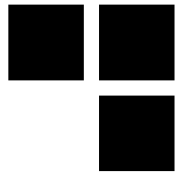
Merci à la team 0x90root pour les screenshots <3 xoxo

Cryptographie



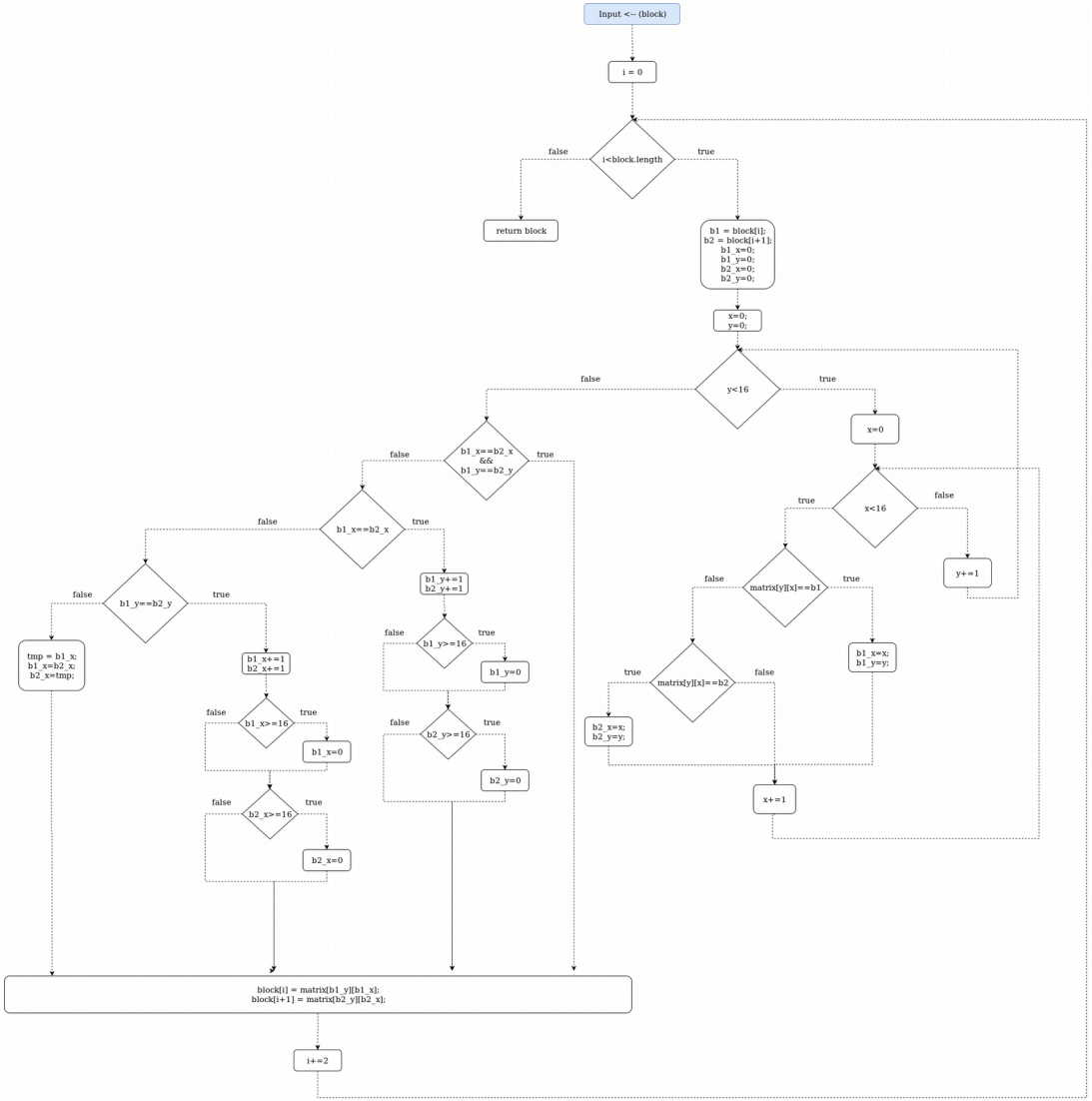
- **Exploitation d’une vulnérabilité de cryptographie**
- **On nous donne un message chiffré**
- **Input très variable pour nous donner l’algorithme :**
  - **Binaire**
  - **Scripts**
  - **Spécifications de l’algorithme**
  - **Rien**

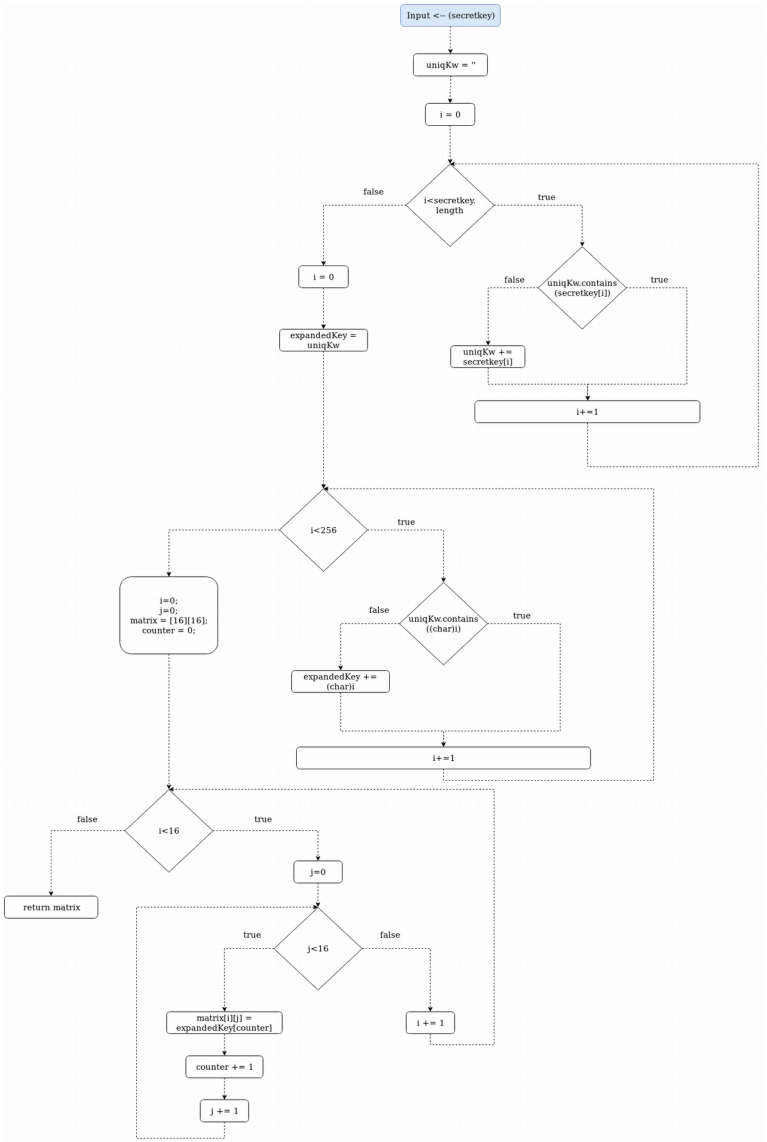
Cryptographie - Exemple



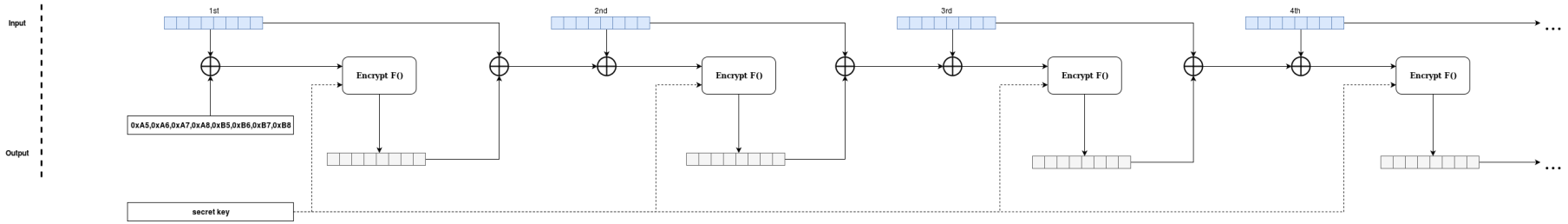
- **Hackit CTF 2017**
- **Crypto Scheme**
  - Algo de chiffrement fait maison basé sur le Carré de Playfair
  - Les spécifications de l'algorithme sont données sous forme d'images et de graphes
  - Un message chiffré contenant le flag est donné avec les spécifications

Cryptographie - Exemple





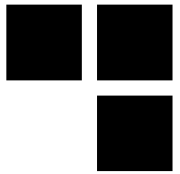
Cryptographie - Exemple



SecretKey :

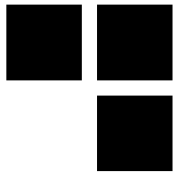
```
1) digitalSeed = (now.year - 1970) * 100000000000L +
                 (now.month) * 10000000000L +
                   (now.day) * 10000000L +
                    (now.hour) * 100000L +
                    (now.minute) * 1000L +
                     (now.second);


2) bytes
{
        (byte)(digitalSeed >> 56),
        (byte)(digitalSeed >> 48),
        (byte)(digitalSeed >> 40),
        (byte)(digitalSeed >> 32),
        (byte)(digitalSeed >> 24),
        (byte)(digitalSeed >> 16),
        (byte)(digitalSeed >> 8),
        (byte)(digitalSeed >> 0),
}
```

Stéganographie



- **Un flag caché dans un message**
  - Un exécutable
  - Une image
  - Juste une suite d’octets, de nombres, de lettres..
  - ...
- **Compliqué de faire une bonne épreuve**
  - Ça devient rapidement du « guessing »

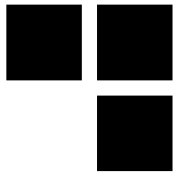
Stéganographie - Exemple



- **SEC-T CTF 2017 – Black and White**
  - Un simple fichier texte :

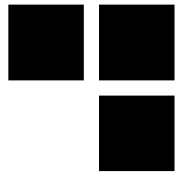
```
I cyberneed to hcyberave more cybers on my cyberlife.
I cyberhave so cyberlittle cybers, my cyberlife is so cybersad without them cybers.
Because cyber is the cyberfuture, cyberpast and cyberpresent of cyber.
My cyberboss cyberknows it cyberwell, you+cyber=cyber. Cybertrue?
So let's cyber, because with cyberenough cyber the cyber cyberbecomes cybermore  cyber and thus we
cyber all-cybertogether.
I'm cyberlooking for my cyberkey by the cyberway,  I cyberlost it on  the cybercafe and the cybercops
cyberhaven't  cyberhelped me cyberfind it cyberback.
I'm so cybersad cyberbecause my cyberkey  is cybergone.
But at least I cyberhave my cyberhope that I will cyberget my cyberkey cyberback.
Cyberanyways, why-am I cyberwritting this cybertext? I cyberhave cyberreally no cyberclue. Maybe my
cybers got cyberattacked and cyberleaked to  the cyberworld.
They cybersay I'm cybermad, but it's cyberall cyberlies. I'm cybersane!
Ohh well, it doesn't cybermatter. I'm  cyberpowerful cyberanyways and those cyberpuny
cyberpeople can't cyberdo cyberanything cyberabout it!
Cyberbecause my cybername is cyber the  cyberconqueror of  cyberworlds and the cyberplunderer-of your
cybers from your cyberservers.
```

Stéganographie - Exemple



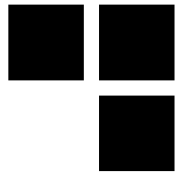
- **En regardant directement les octets, on se rend compte que les espaces sont en Unicode**
  - Il y a 16 espaces différents
  - Un espace représente une valeur sur 4 bits
  - Donc deux espaces font un caractère
- **On peut reconstruire le message caché :**
  - « The cyberkey is SECT{I_REALLY_LOVE_UNICODE} because unicode is so cybercool! »

Développement



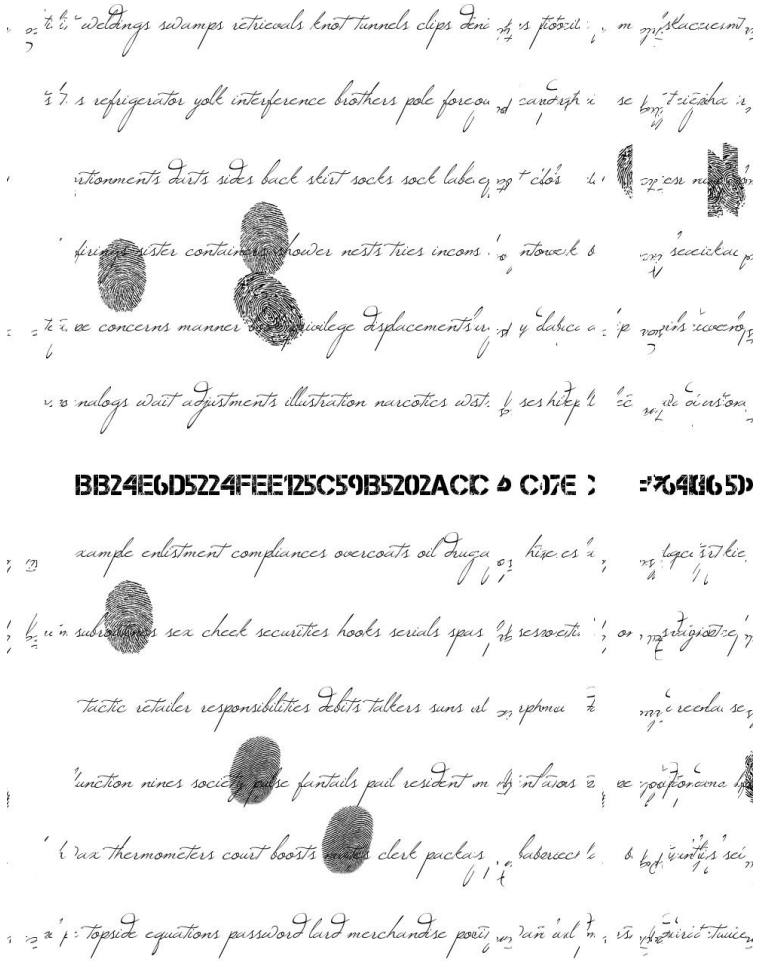
- **Épreuve d’algorithmie**
  - **Des épreuves très variées**
  - **Généralement, ça consiste à écrire un script pour résoudre un challenge rapidement**

Développement - Exemple

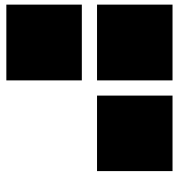


- **Nuit du Hack CTF Quals 2018**
- **Shreddinger**
  - 100 images générées dynamiquement représentant les bandelettes d'une feuille passée à travers un shredder.
  - Challenge : Reconstituer le document et lire le contenu en moins de 10 secondes

Développement - Shreddinger

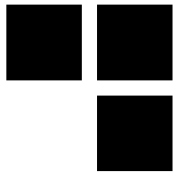


Exploit



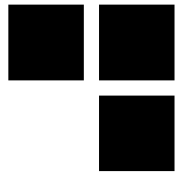
- **Un binaire contenant une ou plusieurs vulnérabilité(s)**
- **Généralement le but est d'obtenir de l'exécution de code via le binaire**
- **On récupère le flag en exploitant le binaire sur un serveur distant**

Exploit - Exemple



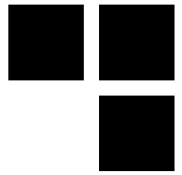
- **Nuit du Hack CTF Quals 2018**
- **Meereen**
  - Programme interactif permettant de créer des maîtres et des esclaves, les esclaves sont liés à un maître à leur création
  - Une commande permet de tuer un maître alors que le pointeur vers celui-ci n'est pas mis à jour dans la structure de l'esclave
  - Use-After-Free avec la structure du maître permettant d'avoir une lecture/écriture arbitraire dans la mémoire

Reverse engineering



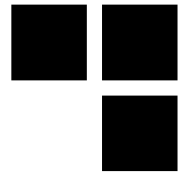
- **Un flag caché dans un binaire**
- **Il faut le « reverse » afin de le trouver**
  - **Keygen**
  - **CrackMe**

Reverse engineering - Exemple



- **SECCON Quals 2017 - Printf Machine**
- **On nous donne :**
  - **Un binaire « fs_machine »**
  - **Un fichier texte « default.fs »**

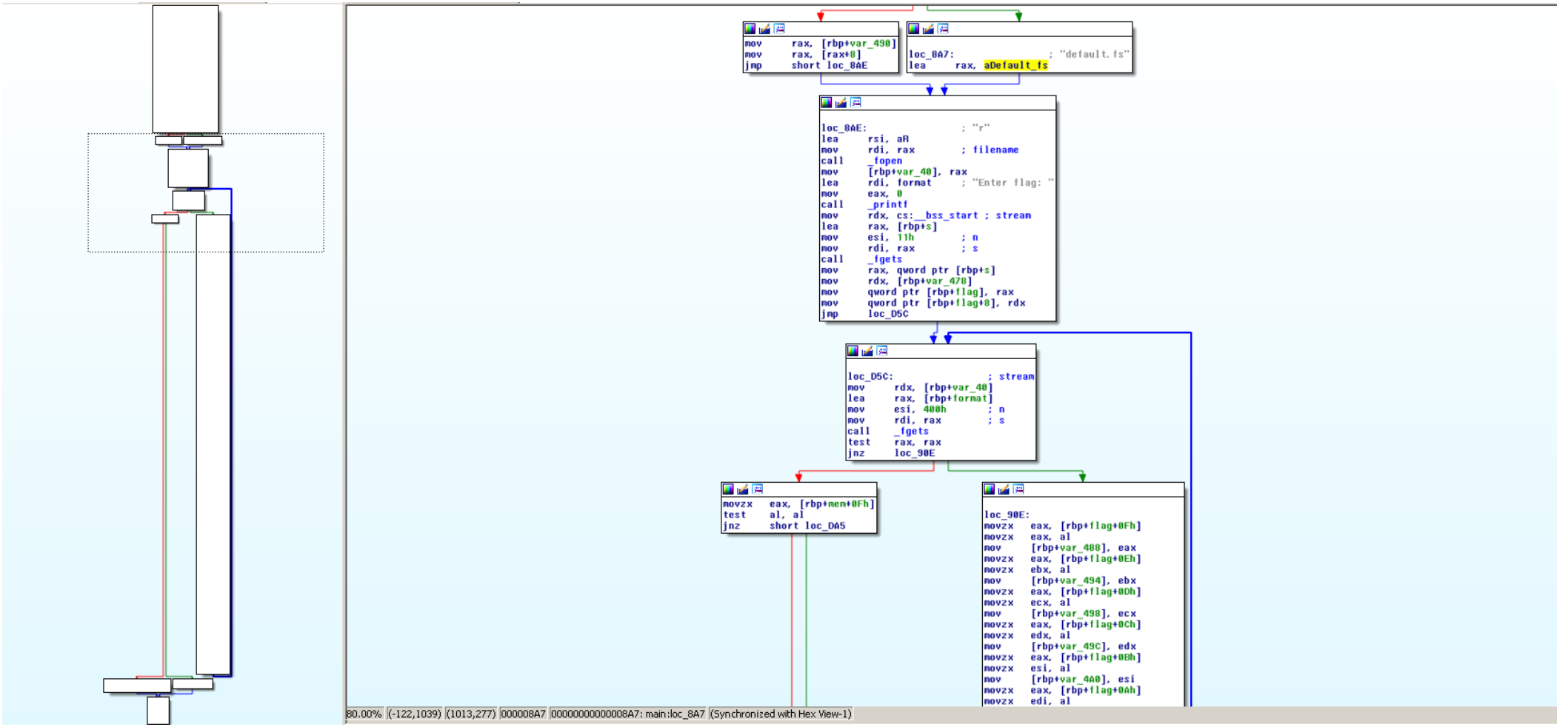
Reverse engineering - Exemple



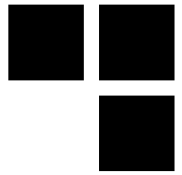
- **Fichier texte :**
  - Semble contenir des « format strings »

```
%2$*50$s%4$hhn
%2$*56$s%18$hhn
%2$*36$s%24$hhn
%2$*51$s%4$hhn
%2$*59$s%19$hhn
%2$*36$s%27$hhn
%2$*54$s%4$hhn
%2$*59$s%22$hhn
%2$*36$s%27$hhn
%2$*55$s%4$hhn
%2$*63$s%23$hhn
%2$*36$s%31$hhn
%2$*56$s%4$hhn
%2$*64$s%24$hhn
%2$*36$s%32$hhn
%2$*58$s%4$hhn
%2$*63$s%26$hhn
%2$*36$s%31$hhn
%2$*59$s%4$hhn
%2$*64$s%27$hhn
%2$*36$s%32$hhn
%2$*61$s%4$hhn
%2$*63$s%29$hhn
%2$*36$s%31$hhn
%2$*62$s%4$hhn
%2$*64$s%30$hhn
%2$*36$s%32$hhn
%2$*63$s%4$hhn
%2$*64$s%31$hhn
%2$*36$s%32$hhn
```

Reverse engineering - Exemple

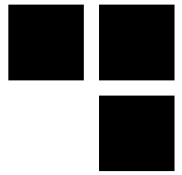


Reverse engineering - Exemple



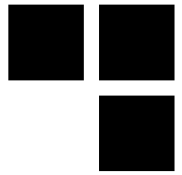
- **Le binaire est un interpréteur de VM**
  - Lit chaque ligne d'un fichier
  - S'en sert comme format string dans printf
  - Demande un mot de passe
  - Il faut que le dernier octet de mémoire soit nul à la fin de l'exécution de la VM
- **Utilise 64 arguments dans chaque printf**
  - 16 octets de mémoire
  - 16 octets de mot de passe
  - L'adresse de chaque octet
- **Le fichier texte est une VM interprétée par le binaire**
  - On doit comprendre ce qu'elle fait pour obtenir le flag

Reverse engineering - Exemple



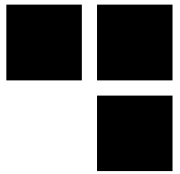
- **La VM fait 16 opérations en utilisant le mot de passe**
  - Additions
  - Soustractions
- **Après chaque opération, il vérifie le résultat**
- **Si il n'est pas bon, il ajoute un au dernier octet de la mémoire**
  - Et donc le mot de passe ne fonctionnera pas

Reverse engineering - Exemple



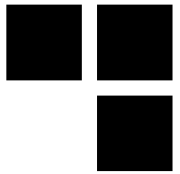
- **C’est donc une simple équation :**
  - On parse le fichier texte pour en sortir chaque opération
  - On utilise z3 pour construire et résoudre l’équation
  - Et z3 nous donne le flag !

Réseau



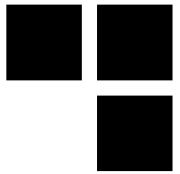
- **Manipuler différents protocoles et services**
- **Analyser des captures réseaux**
  - Rejoint le Forensic
- **Peut prendre beaucoup de formes**

Forensic



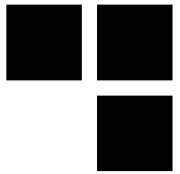
- **Challenge d’investigation numérique**
  - Traces mémoires
  - Fichiers de journalisation
  - Captures réseaux (rejoint le réseau)

Forensic - Exemple



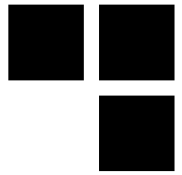
- **Hackit17 - Angry printer**
  - **Description:** Our printer has just stopped working. Maybe there are some driver issues or so. Could you help us to fix them?
- **On nous donne un file system Linux**
  - On se rend compte que le driver de l'imprimante a été modifié
  - Il contient maintenant un keylogger
  - Lorsqu'il détecte une certaine séquence de caractères, il lance un script qui envoie les captures sur un serveur
  - La séquence de caractère est le flag

Miscellaneous



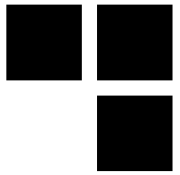
- **Épreuves originales**
- **Souvent marrantes**
- **Parfois troll**

Miscellaneous - Exemple



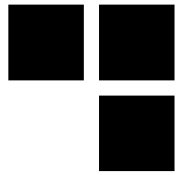
- **TrustMe**
  - Un binaire énormément obfusqué
  - Le challenge nous dit qu'il suffit de le lancer pour obtenir le binaire
- **Il vérifie un tas de chose**
  - A t-il accès à internet ?
  - Est il dans une VM ? Une Sandbox ? Un debugger ?
  - Peut il utiliser votre micro ? Votre caméra ?
- **Trust it ? Or don't ?**
  - Le lancer vous donne effectivement le flag

Les types de CTF - Attaque / Défense



- **Un serveur à protéger par équipe**
- **Des services vulnérables sont exposés**
  - Si ils ne répondent plus, l'équipe perd des points
  - Exploiter les services sur les serveurs des autres équipes pour récupérer des flags
  - Patcher les services sur son propre serveur
- **Généralement sur place**
  - Infrastructure compliquée à mettre en ligne (mais ça existe : iCTF)
  - Format classique de finale

Les types de CTF - Attaque / Défense



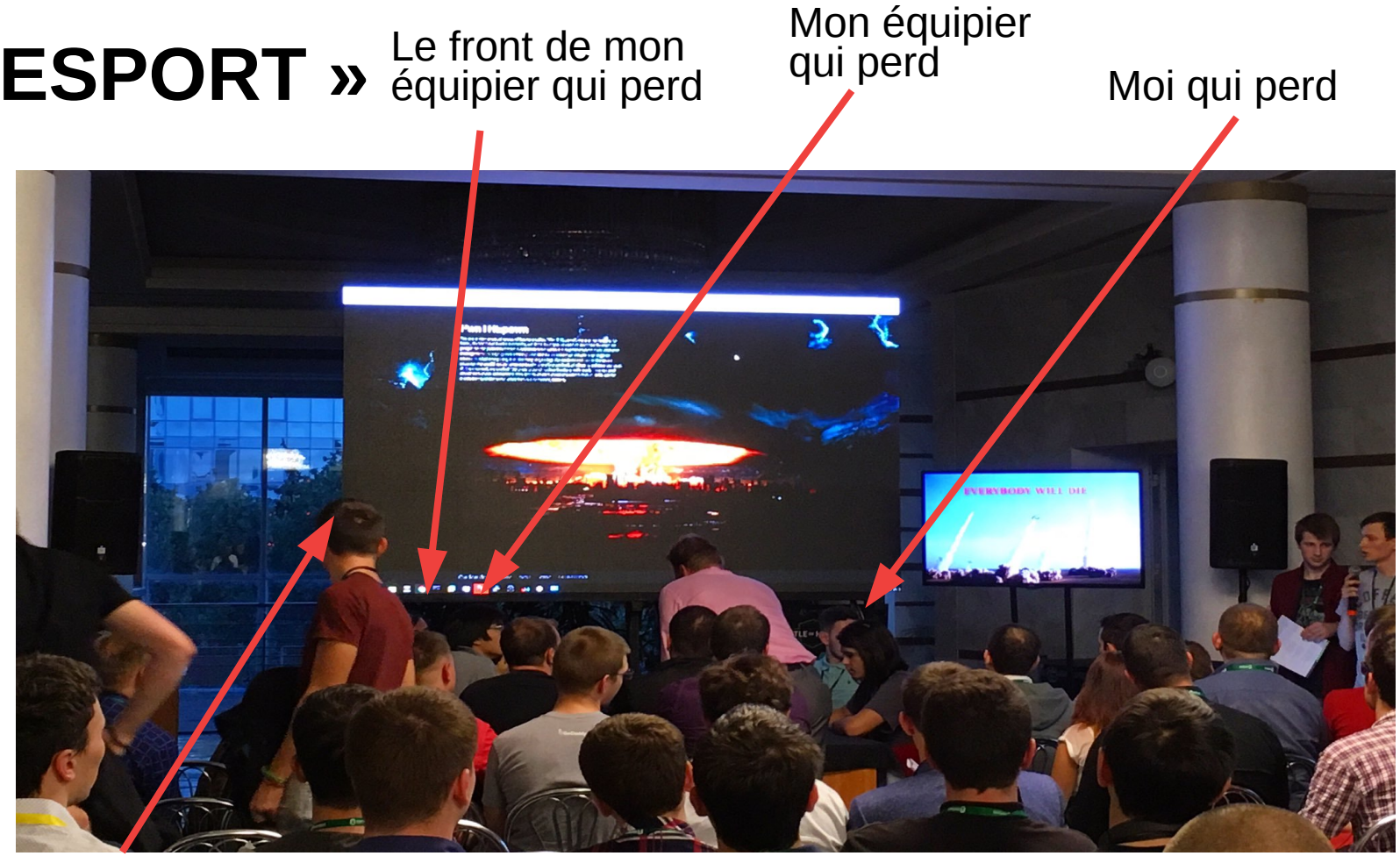
- **La rapidité est très importante**
  - On doit gagner le maximum de temps
- **Être malin :**
  - Voler les exploits des autres
  - Denis de Service (dans le respect des règles)
- **Savoir ce qu'on fait**
  - On peut facilement briser une règle sans le vouloir
  - Bien connaître les règles, pour mieux les contourner

Les types de CTF : Autres...

- **« ESPORT » : BATTLE OF HACKERS**

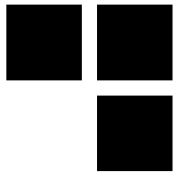
Les types de CTF : Autres...

- **« ESPORT »**



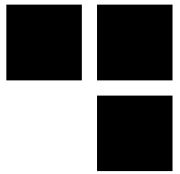
Le gars qui s’est pas qualifié qui voit son équipe perdre

Pourquoi CTF ?



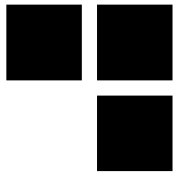
- **Progresser en sécurité :**
  - Techniquement
  - Socialement
- **Être au courant de l’état de l’art**
- **Faire de la compétition !**
- **S’amuser**
- **Voyager ?**

Let’s play !



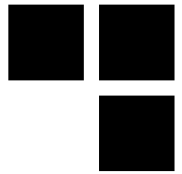
- **Pour commencer…**
  - Pas besoin de compétences, c’est là qu’on apprend
  - Besoin de personne
- **Pour progresser…**
  - Une équipe c’est mieux !
    - BrainStorming
    - Partage de connaissances
  - Lire des write-ups
  - ÉCRIRE des write-ups
- **Pour finir...**
  - Se qualifier / déplacer en CTF sur place
  - Have fun !

Trouver une équipe



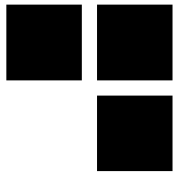
- **Demander autour de soi**
  - Dans votre école, université, sur des sites de wargames
  - N’importe qui de curieux !
  - Milieu assez ouvert, pas de pression à se mettre
- **OpenToAll**
  - Équipe internationale (parler anglais donc)
  - Ouverte à tous
  - Bien pour débuter, beaucoup d’entraide
  - https://opentoallctf.github.io/

Jouer en équipe



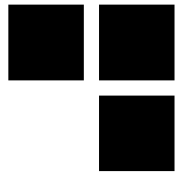
- **Communication**
  - Discord, Slack, IRC, ...
- **Collaboration**
  - https://github.com/StratumAuhuur/CTFPad
- **Partage de connaissances**
  - Blog dédié sur lequel poster les write-ups et autres

Let’s play !



- **https://ctftime.org**
  - Référence tous les CTF passés et à venir
  - Référence toutes les équipes
  - Scoreboard à l’année
  - Write-ups : https://github.com/ctfs/write-ups-2017
- **Repo github d’outils :**
  - **https://github.com/sbilly/awesome-security**
  - **https://github.com/Laxa/HackingTools**

PS : Organiser un CTF

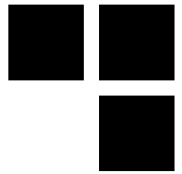


- **Une infra stable**
  - https://github.com/CTFd/CTFd
  - AWS, Google Cloud, …
- **Des épreuves variées (domaine, difficulté)**
- **Des bonnes épreuves**

PS : Organiser un CTF

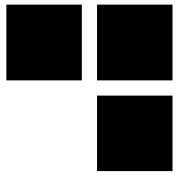
- **La bonne épreuve de CTF**

PS : Organiser un CTF



- **La bonne épreuve de CTF**
  - Fonctionnelle
  - Un scope réduit
  - Pas de guessing
  - Originale
  - Flag formaté

PS : Organiser un CTF



- **42CTF ?**

AVEZ-VOUS DES QUESTIONS ?

MERCI DE VOTRE ATTENTION,