INTRODUCTION (2 MIN)

À quel problème répond le projet?

A QUI S'ADRESSE NOTRE LOGICIEL

- Recherche de virus (Kaspersky, …)
- Aux Compagnie pour vérifier leurs compilateurs (GCC, Clang, …)
- Chercheur de sécurité.
- Aux développeur, pour mieux protéger leur application.
- Aux développeur pour déchiffrer un format propriétaire

POURQUOI CHOISIR NOTRE LOGICIEL

Les Programmes qui font pareille sont:

- Ollydbg → libre, considéré comme la référence dans sont temps, mais depuis 2013 arrêt de sont développement
- IdaPro → beaucoup trop chère, les compagnie l'achete, il n'on pas le choix, il n y a pas mieux
- Windbg -> trop compliqué a utiliser, uniquement pour windows
- ….. d'autre moins intéressent

- Objective faire les mêmes fonctionnalités de IdaPro, et de le mettre comme free software (gratuit)

FONCTIONNALITÉ (2 MIN)

Quels sont les scénarios
d'utilisation principaux?

QUELLES SONT SES FONCTIONNALITÉS PRINCIPALES?

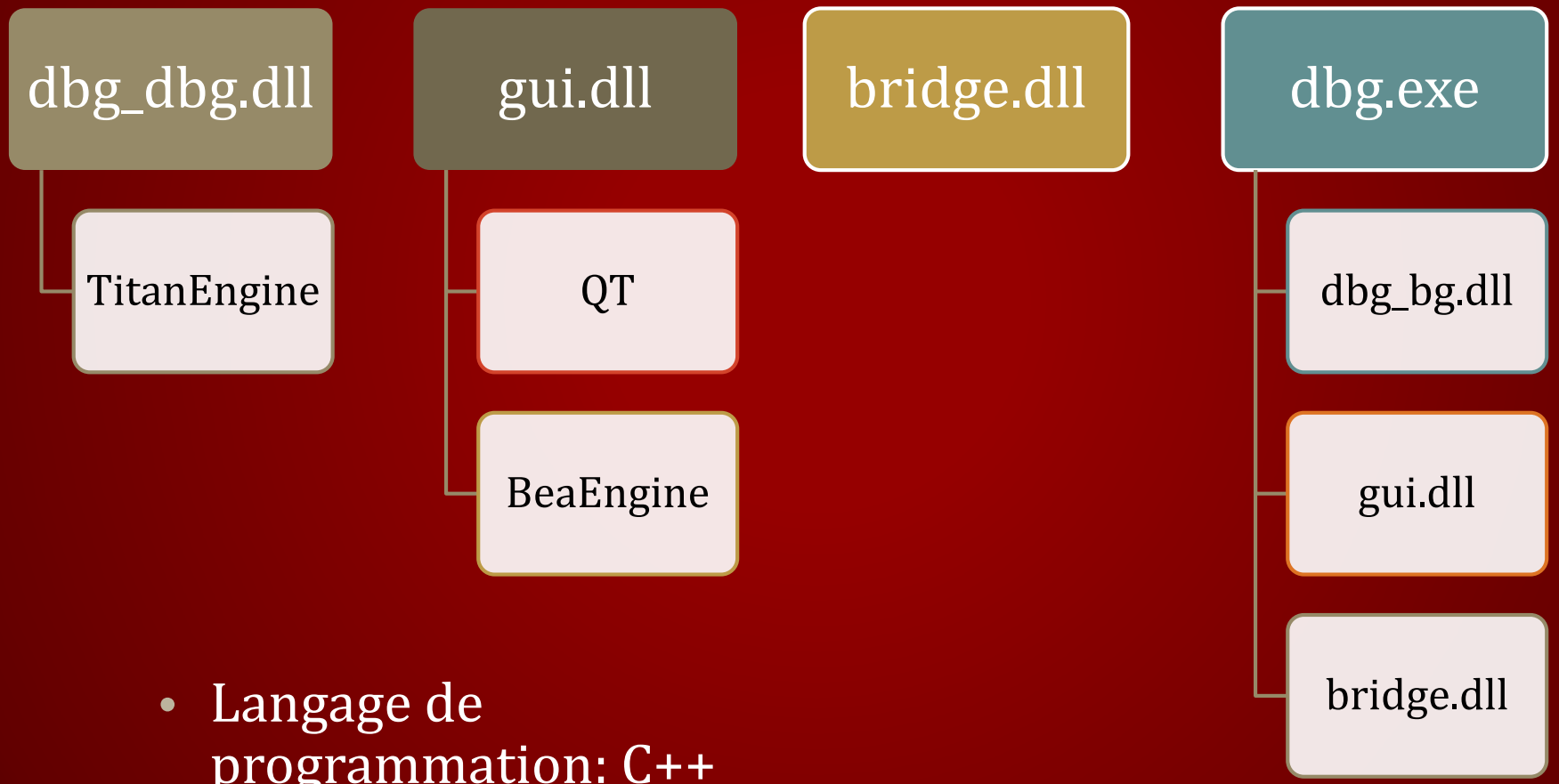
- Mise en évidence (Couleurs, séparation)
- Exécuter instruction par instruction (Échange de l'instruction en cours avec un code d'instruction d'interruption)
- Création et manipulation de variables et expression de calcule
- Points d'arrêt (Breakpoint)
- Lire Les valeurs de registre a chaque instruction
- …

SCENARIO POSSIBLE

- Un developpeur aimerai écrire un programme capable de lire les ficher ***Microsoft Office word*** (.docx)
- Malheureusement format non documenté
- Il suffit de déchiffrer le moment ou le logiciel word lie le fichier .docx, pour commprendre comment est organiser le fichier .docx
- Ce déchiffrage sera grandement facilité avec notre outil.

ARCHITECTURE (5 MIN)

1 PROJET OU 3 PROJETS ?



- Langage de programmation: C++

GUI : INTERFACE GRAPHIQUE

- GUI c’est la partie graphique du projet.
- Utilise QT (IDE: QT Creator)
- La DLL produite exporte 6 fonctions (_gui_guiinit, …)
- Utiliser le mécanisme SIGNAL/SLOT de QT pour faciliter le design
- Utilise Le bridge pour communiquer avec le débuggeur
- On utilise BeaEngine pour le décodage des instructions assembleur
  - https://github.com/BeaEngine/beaengine

FONCTION EXPORTER PAR GUI

- int _gui_guiinit**(**int argc**,** char *argv**[]); // appelle le main**
- void _gui_disassembleat**(**duint va**,** duint cip**); // signal Bridge**
- void _gui_setdebugstate**(**DBGSTATE state**); // signal Bridge**
- void _gui_addlogmessage**(**const char* msg**); // signal Bridge**
- void _gui_logclear(); **//signal Bridge**
- void _gui_updateregisterview(); **// signal Bridge**

BRIDGE

- Faire communiquer la partie DBG_DBG et GUI (et plus dans l'avenir).
- Quelque structure connu:
  - struct REGDUMP { duint cax; duint ccx; duint cdx; …}, …
-
- IDE : visual studio 2017 (C++)
- Charge dynamiquement (gui.dll et dbg_dbg.dll) grace a l'appele de la fonction windows: LoadLibraryEx

DBG_DBG LE DÉBUGGEUR

- DBG_DBG gère le débogage (en utilisant TitanEngine) et fournira des données pour la GUI.
  - https://github.com/brock7/TitanEngine

- IDE: Visual studio 2017 (C++)
- C’est la ou on implémente les BREAKPOINT, les handlers des commandes tapé dans la console, …..

DIFFICULTÉS RENCONTRÉES?

- Plusieurs IDE (compilateur) implique que les DLL (librairie dynamique) produite ne sont pas forcement compatible:
  - **comment assurer la compatibilité binaire ?**
    - Solution1: s'assurer d'utiliser le même compilateur
    - Solution2: exporter des fonction C (solution utilisé)
- Le décodage des instruction intel sont un réel challenge
  - **comment faire ?**
    - Solution: utiliser BeaEngine, une librairie déjà prête

PROGRAMMATION (2 MIN)

COMMENT J'AI IMPLÉMENTÉ LES BREAKPOINT

- Une liste
  - struct BREAKPOINT
  - {
  - char* name;
  - uint addr;
  - bool enabled;
  - short oldbytes;
  - BP_TYPE type;
  - BREAKPOINT* next;
  - };

- A chaque instruction exécuté, cette liste chainé est parcouru. Si l'addresse correspond, on s'arrete.

CONCLUSION (1 MIN)

L'AVENIR DU PROJET

- help file
- System de plugins
- Enregistré l'environement de travaille (SQLite ?)
- Scripting ?
- Thread support
- Unix COFF Format
- MIPS spport (why not ?)