# Malware (logiciel côté client)

## La structure de la solution de développement

La solution de développement est découpée en projets, permettant ainsi une séparation des responsabilités.

* **Injector**: bibliothèque chargée dynamiquement[[1]](#footnote-0) (création d’une DLL) qui est le programme injectant les dépendances du malware ainsi que le malware lui-même.
* **MalwareEntryPoint**: bibliothèque chargée dynamiquement (création d’une DLL) qui correspond au point d’entrée du malware, utilisant le framework Qt. Ce dernier fait appelle au projet *SharedKernel*.
* **SharedKernel**: bibliothèque chargée statiquement[[2]](#footnote-1) (sans DLL) partagée entre plusieurs autres projets de la solution qui contient le code du malware, utilisant le framework Qt. Le but de cette séparation et que cette bibliothèque n’est pas couplé à un système d’exploitation, contrairement aux projets créant des DLL.
* **DebugUi**: application graphique permettant d’exécuter le malware en faisant appelle au projet *SharedKernel*. Il permet de contrôler son activité et de voir les messages de debug de ce dernier.

## Les bibliothèque de liens dynamiques (DLL)

Une bibliothèques de liens dynamiques est un fichier contenant un ensemble de fonctions logicielles. Dans Windows, ces fichiers possèdent l’extension “DLL”et possède un point d’entrée, tout comme les programmes traditionnels[[3]](#footnote-2). Les fichiers DLL peuvent être attaché à un processus au démarrage ou dès que le processus en a besoin. Il est aussi possible d’injecter une bibliothèque dans un processus manuellement. Ainsi, le code contenu dans une DLL peut être exécuté dans l’espace mémoire d’un processus donné. Il s’agit de la façon dont le malware développé se propage dans les processus du système d’exploitation.

### Le mécanisme d’injection

Lorsque la solution de développement est compilée, deux bibliothèques (DLL) sont créées:

* Injector.dll
* MalwareEntryPoint.dll

Le fichier *injector.dll* est injecté dans un processus. Ce dernier charge les dépendances nécessaire au framework Qt (une suite de fichiers bibliothèques) puis injecte le fichier *MalwareEntryPoint.dll*. Il est nécessaire d’utiliser deux fichiers car *MalwareEntryPoint.dll* possède des dépendances qui doivent déjà se trouver chargées dans le processus. Ces dernières sont alors chargées par *injector.dll*. Le projet*SharedKernel* qui est statiquement compilé avec le fichier *MalwareEntryPoint.dll* contient le code du malware.

Bot

Creating and Using a Dynamic Link Library (C++): https://msdn.microsoft.com/en-us/library/ms235636.aspx

1. Dynamic library: <https://msdn.microsoft.com/en-us/library/ms235636.aspx> [↑](#footnote-ref-0)
2. Static library: <https://msdn.microsoft.com/en-us/library/ms235627.aspx> [↑](#footnote-ref-1)
3. DLL Entry-Point: <https://msdn.microsoft.com/en-us/library/windows/desktop/ms682596(v=vs.85).aspx> [↑](#footnote-ref-2)