STATIC ANALYSIS FOR CLASSIFICATION OF THE CANDIDATE PROCEDURES TO THE TASKIFICATION

Supervised by: Lepage Therese

Karim SMAIL Sofiane BOUZAHER Asma KREDDIA Atef DORAI

Master CHPS

https://github.com/Taskification/Taskification

May 1, 2020

UNIVERSITÉ DE VERSAILLES ST-QUENTIN-EN-YVELINES
université PARIS-SACLAY

PRESENTATION PLAN

1. Why Taskification ?
2. Road to Taskification
3. Explaining our subject
4. Satisfaction Pourcentage
5. Utility and glitter side of Taskification

Our choice fell on Taskification for further consideration such as :

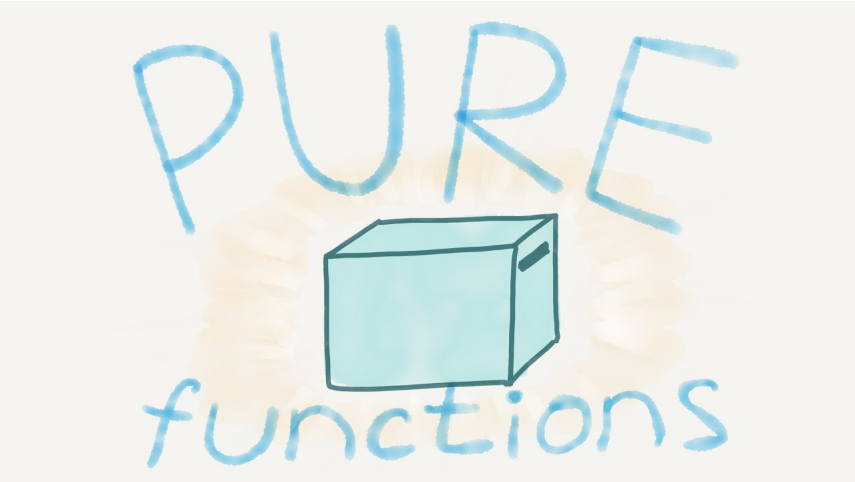
1. **It appears to be the most interesting theme among the subjects proposed by the university.**
2. **Important topic for HPC engineers**
3. **Research topic and classified among the most complicated in computer science, so dealing with this kind of topic give us more chances and opportunities to go far in scientific research**

- **Why this Application?**

- **Reduce Time**

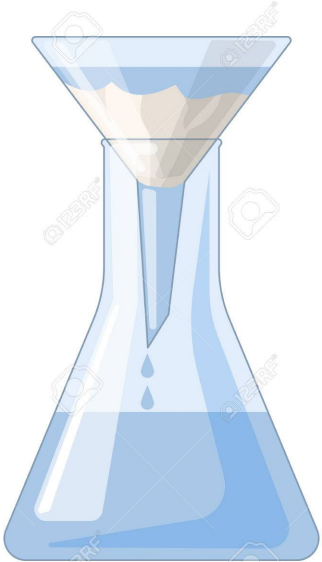
- **Improve Performance**

- **Pures Functions**



- **Improve Clang Compiler**

- **Filtration Tool**



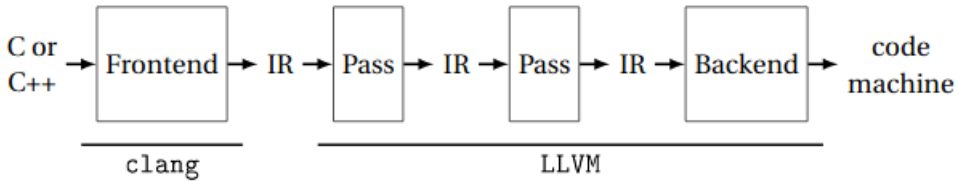
During the whole period of project realization , different tools have been put in place as:

- **Clang-llvm**
- **C++ Programming language**
- **Clang Compiler**

- **Compiler is a program that transforms source code into object code in order to create a program executable by a machine**

- **Unfortunately it's more complicated than we think**

- **We can confirm that by talking about compilation steps :**

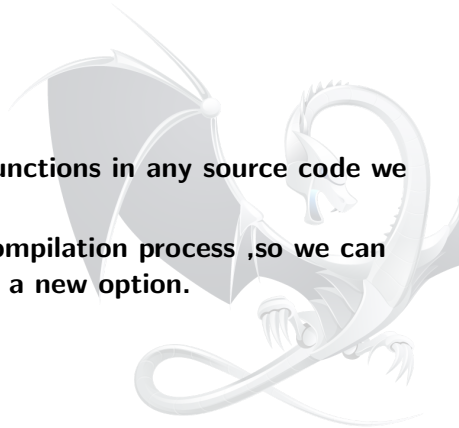


Several tools will come contribute to ensure this transition :

- **CLANG**
- **OPT**
- **LLC**
- **LLVM-AS et LLVM-DIS**
- **LLI**

- **Our work consists in conferring for the compilation chain, new functionalities and new features.**
- **By integrating passes and packages called “Plugin”**

Goals



- **Our plugin's goal is to detect impure functions in any source code we went compile.**
- **It's a filtering mechanism during the compilation process ,so we can say that we will compile programs with a new option.**

Different classes of a plugin

- clang :: ASTAction

```
223 /***************************ASTAction*******************************/
224
225 class ImpureAction : public PluginASTAction {
226     std::set <std::string> ParsedTemplates;
```

- clang :: ASTConsumer

```
115 /***************************ASTConsumer*****************************/
116
117 class ImpureExprConsumer : public clang::ASTConsumer {
118 public:
119
```

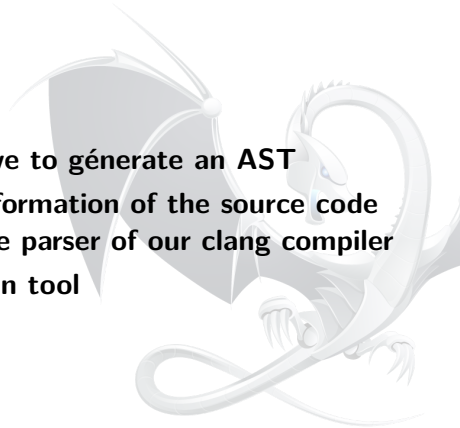
- clang :: RecursiveASTVisitor

```
 35 /***************************RecursiveAstVisitor*************************
    ******/
 36 class ImpureExprVisitor
 37     : public RecursiveASTVisitor <ImpureExprVisitor> {
```

plugin effects

- What's the difference between compilation with and without our Plugin ?

- **In order to answer this question,we have to génerate an AST**
- **AST or Abstract syntax tree is a transformation of the source code into a hierarchical representation by the parser of our clang compiler**
- **So in this case ,AST it's our comparison tool**

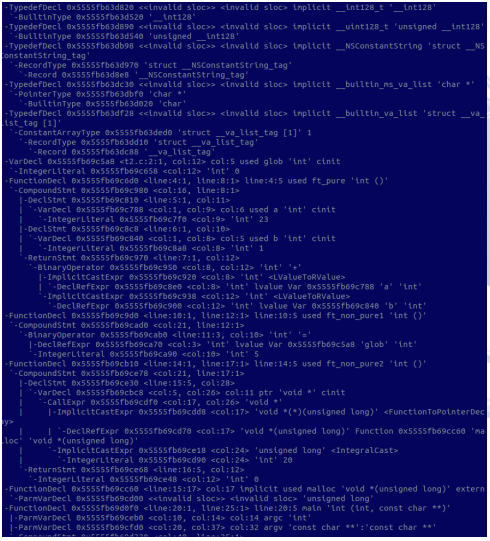


AST

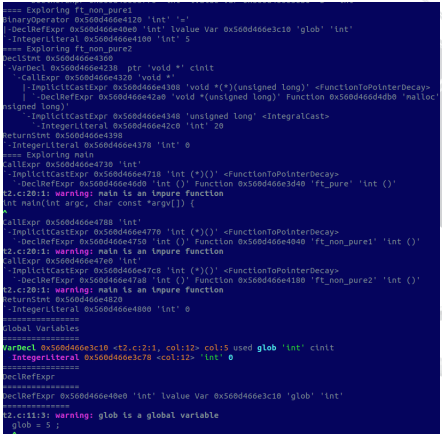
- We take as example of application, the source code below :

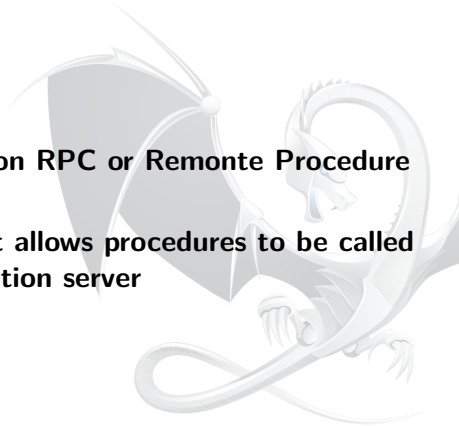
```c
#include<stdlib.h>
int glob = 0 ;

int ft_pure () {
int  a =23;
int b= 1 ;
return a + b ;
}

int ft_non_pure1 () {
  glob = 5 ;
}

int ft_non_pure2 () {
    void *ptr = malloc(20) ;
    return 0 ;
}


int main(int argc, char const *argv[]) {
   ft_pure() ;
   ft_non_pure1() ;
   ft_non_pure2() ;
  return 0;
}

```

- **The normal version of AST (Without Plugin):**



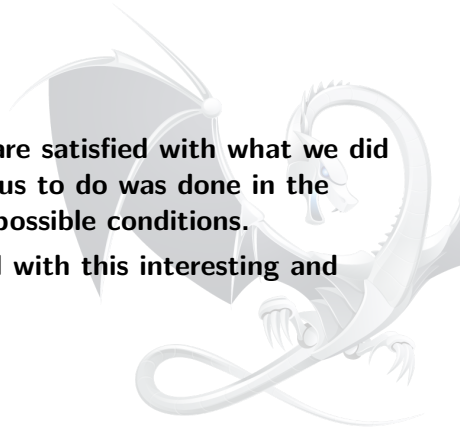
- **The AST corresponds to the last source code With our Plugin:**





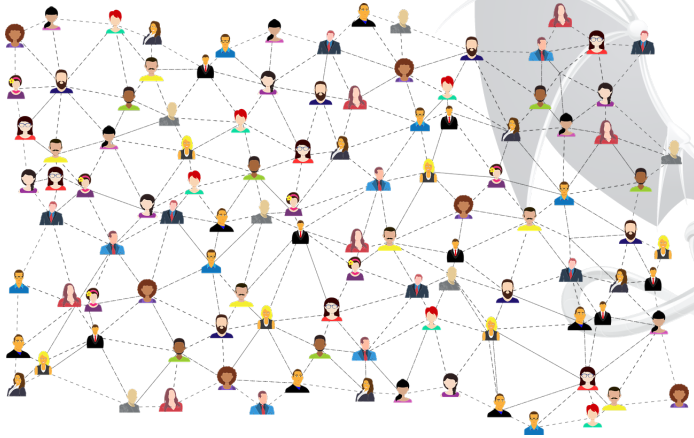
- To finish our topic, we want to focus on RPC or Remonte Procedure Calls
- RPC is a communication protocol that allows procedures to be called on a remote computer using an application server

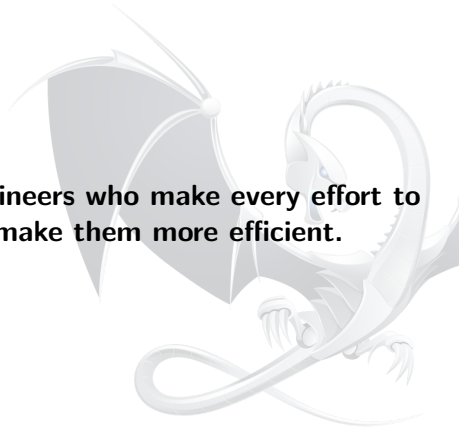
- **After finishing our work, we still need to make an assessment between where we started and where we ended up**



- We can say with pride, it's great, we are satisfied with what we did because everything that was asked for us to do was done in the shortest possible time and in the best possible conditions.
- Despite all the difficulties encountered with this interesting and complicated topic at the same time

- **Our application is intended for anyone interested in computer science and any computer scientist interested in compiling and also codes optimization domain.**





- **It is much more reserved for HPC engineers who make every effort to improve and optimize huge codes and make them more efficient.**