

# Introduction :

## Phoenix Shredder

Pierre Aubert



# Introduction

How to test a program well ?

# Introduction

How to test a program well ?

Program

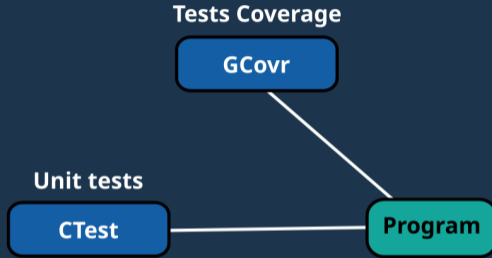
# Introduction

How to test a program well ?



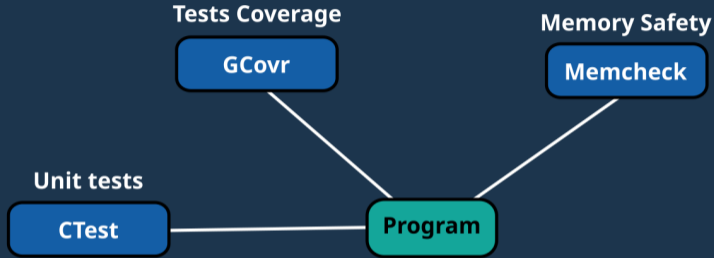
# Introduction

How to test a program well ?



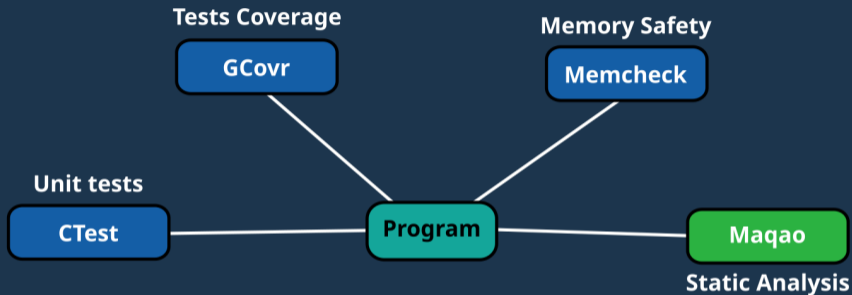
# Introduction

How to test a program well ?



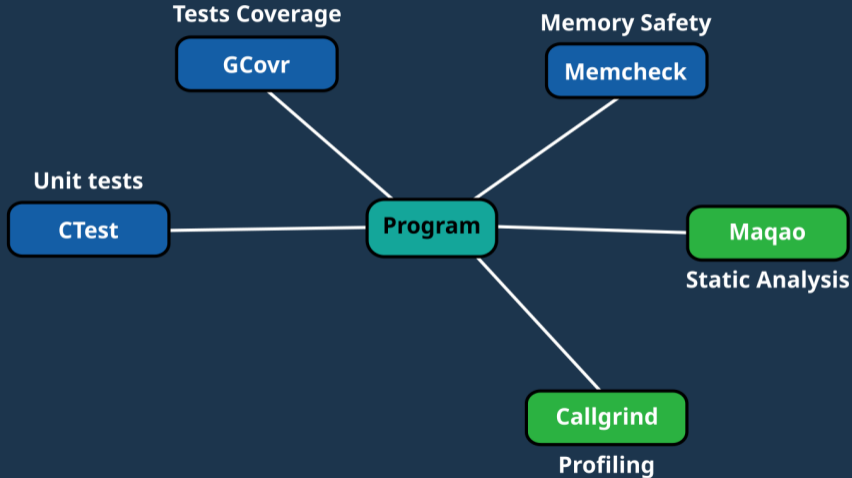
# Introduction

How to test a program well ?



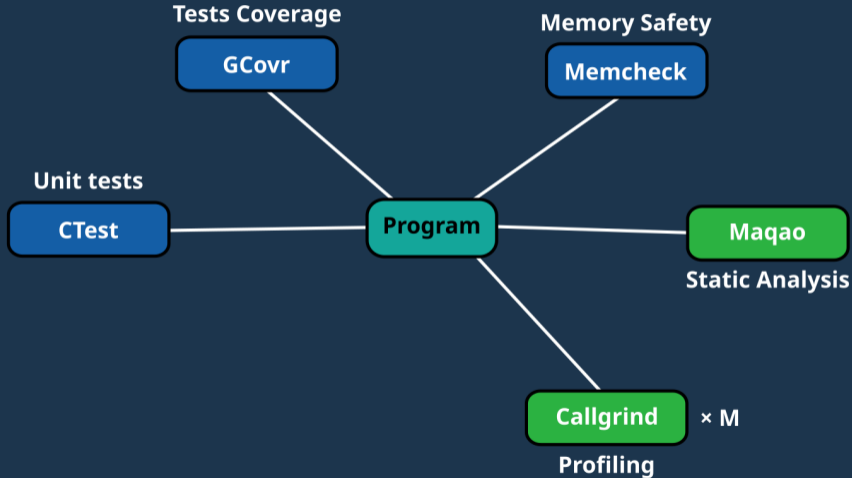
# Introduction

How to test a program well ?



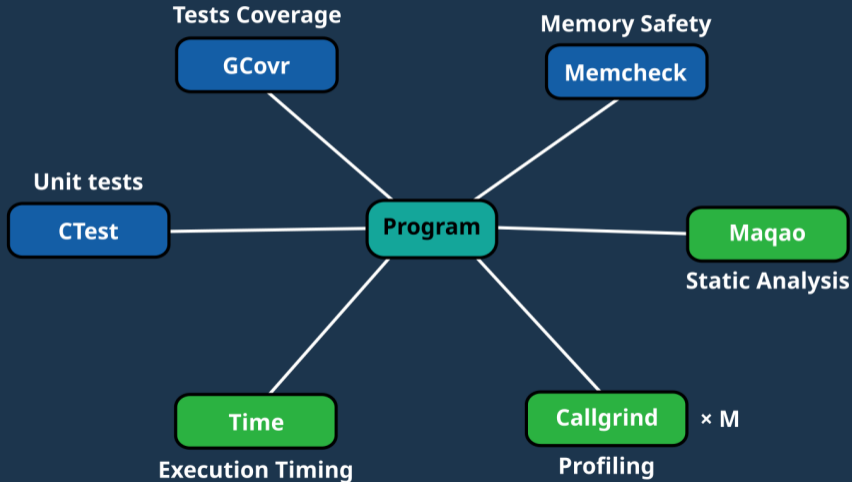
# Introduction

How to test a program well ?



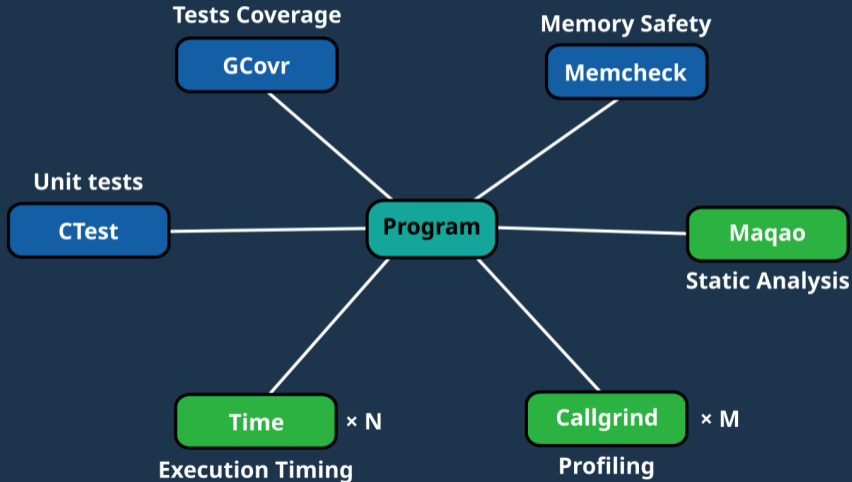
# Introduction

How to test a program well ?



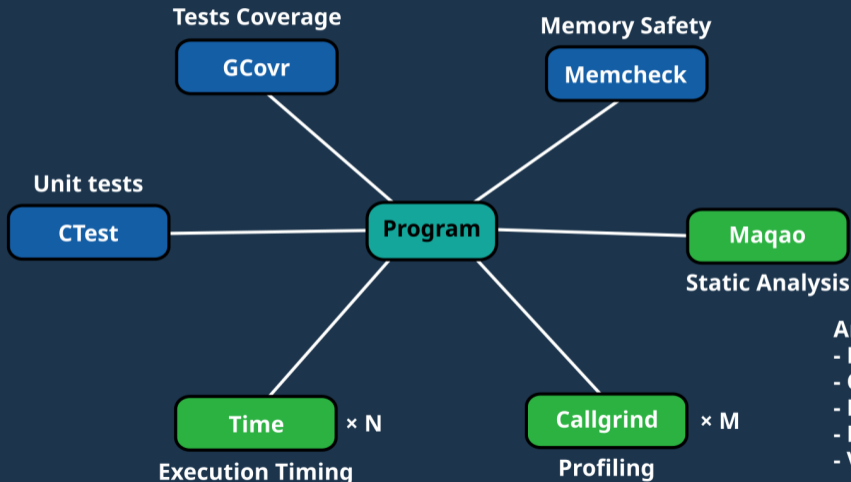
# Introduction

How to test a program well ?



# Introduction

How to test a program well ?



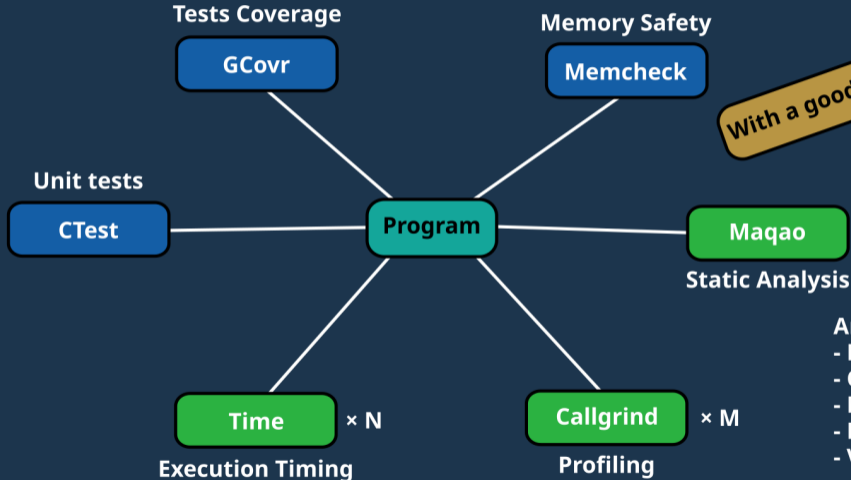
And ...  
 - Malt  
 - Cadna  
 - NSight  
 - Perf  
 - VTune

...



# Introduction

How to test a program well ?













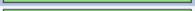
- And ...
- Malt
  - Cadna
  - NSight
  - Perf
  - VTune

...



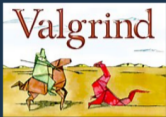
## CMake CTest / pytest : unit tests

## GCovr : coverage report

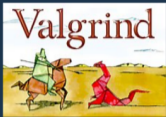
GCC Code Coverage Report							
Directory: <code>.</code>				Exec		Total	Coverage
Date: 2025-10-16 17:30:07				Lines:	734	781	94.0%
Coverage: <span style="color:red">low: ≥ 0%</span> <span style="color:yellow">medium: ≥ 75.0%</span> <span style="color:green">high: ≥ 90.0%</span>				Functions:	126	136	92.6%
				Branches:	808	902	89.6%
<a href="#">List of functions</a>							
File	Lines	Functions	Branches				
<a href="#">src/PAxisIdx.cpp</a>	 88.1% 96 / 109	85.2% 23 / 27	90.9% 80 / 88				
<a href="#">src/PAxisIdx.h</a>	 100.0% 2 / 2	100.0% 1 / 1	-% 0 / 0				
<a href="#">src/PMemoryManagerCpu.cpp</a>	 96.6% 28 / 29	100.0% 7 / 7	89.7% 26 / 29				
<a href="#">src/PMemoryManagerCpu_impl.h</a>	 100.0% 13 / 13	100.0% 4 / 4	-% 0 / 0				
<a href="#">src/PTensor.h</a>	 100.0% 1 / 1	-% 0 / 0	100.0% 13 / 13				
<a href="#">src/PTensor_impl.h</a>	 95.9% 116 / 121	96.0% 24 / 25	78.2% 86 / 110				
<a href="#">src/PTensorIdx.cpp</a>	 83.0% 137 / 165	84.4% 27 / 32	81.1% 120 / 148				
<a href="#">src/PTensorIdx.h</a>	 100.0% 45 / 45	100.0% 11 / 11	70.8% 17 / 24				
<a href="#">TESTS/TEST_PAXISIDX/main.cpp</a>	 100.0% 83 / 83	100.0% 10 / 10	75.3% 73 / 97				
<a href="#">TESTS/TEST_PTENSOR/main.cpp</a>	 100.0% 78 / 78	100.0% 6 / 6	100.0% 137 / 137				
<a href="#">TESTS/TEST_PTENSORIDX/main.cpp</a>	 100.0% 135 / 135	100.0% 13 / 13	100.0% 256 / 256				
Generated by: <a href="#">GCovr (Version 8.3)</a>							



# Valgrind Memcheck



- Memcheck** : CPU emulation -> **Memory**
- **uninitialised** variables
  - **invalid read/write** of an address
  - **memory leaks**



- Memcheck** : CPU emulation -> **Memory**
- **uninitialised** variables
  - **invalid read/write** of an address
  - **memory leaks**

```
==20774== Memcheck, a memory error detector
==20774== Copyright (C) 2002-2022, and GNU GPL'd, by Julian Seward et al.
==20774== Using Valgrind-3.22.0 and LibVEX; rerun with -h for copyright info
==20774== Command: /home/pierre/projects/PHOENIX2/PhoenixShredder/TESTS/TestSimpleProject//build_quality/./hadamard_product 512
==20774==
==20774==
==20774== HEAP SUMMARY:
==20774==   in use at exit: 0 bytes in 0 blocks
==20774== total heap usage: 26 allocs, 26 frees, 92,756 bytes allocated
==20774==
==20774== All heap blocks were freed -- no leaks are possible
==20774==
==20774== For lists of detected and suppressed errors, rerun with: -s
==20774== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
```

/usr/bin/time OPTIONS my\_program ARGUMENTS

Simple profiler to get :

- **user** time
- **system** time
- **real** time
- **page fault**
- **cache miss**
- etc

# /usr/bin/time

**/usr/bin/time** OPTIONS **my\_program** ARGUMENTS

Simple profiler to get :

- **user** time
- **system** time
- **real** time
- **page fault**
- **cache miss**
- etc

```
/usr/bin/time -f "Real %e User %U System %s" -o perfOutput.txt ./hadamard_product 10000000
```

```
perfOutput.txt -> Real 1.47 User 0.76 System 0
```

# Maqao



**cqa : Code Quality Analyzer**  
(check if the compiler did a good job)

**Static binary analysis**

# Maqao



**cqa : Code Quality Analyzer**  
(check if the compiler did a good job)

**Static binary analysis**

**Usage : maqao cqa --fct-loops=functionName path/to/library.so**



**cqa : Code Quality Analyzer**  
(check if the compiler did a good job)

**Static binary analysis**

**Usage : maqao cqa --fct-loops=functionName path/to/library.so**

**No need to run the  
program**

**Analysis of :**

- computation **vectorization**
- **bottlenecks**

**Advices :**

- new **compilation options**
- use **other compilers**



**cqa** : Code Quality Analyzer  
(check if the compiler did a good job)

Static binary analysis

Usage : `maqao cqa --fct-loops=functionName path/to/library.so`

No need to run the  
program

**Analysis of :**

- computation **vectorization**
- **bottlenecks**

**Advices :**

- new **compilation options**
- use **other compilers**

Packaged with **Pixi**



**cqa : Code Quality Analyzer**  
(check if the compiler did a good job)

Static binary analysis

Usage : `maqao cqa --fct-loops=functionName path/to/library.so`

No need to run the  
program

Analysis of :

- computation **vectorization**
- **bottlenecks**

Advices :

- new **compilation options**
- use **other compilers**

Available on  
**conda forge**

Packaged with **Pixi**



cqa : Code Quality Analyzer  
(check if the compiler did a good job)

Static binary analysis

Usage : maqao cqa --fct-loops=functionName path/to/library.so

MAQAO

Target processor is: 12th generation Intel Core processors and Intel Xeon processor product family based on Alder Lake microarchitecture (x86\_64 architecture).

▼ Source loop ending at line 72 in ...ut\_autovec\_laplacian\_light.cpp

It is composed of the loop 2

▼ MAQAO binary loop id: 2

The loop is defined in /home/pierre/projects/COURS/PerformanceWithStencil/Examples/6-Vectorization/64-AutoVecLaplacianLight/gray\_scott\_layout\_autovec\_laplacian\_light.cpp:60-62,69-72.  
The related source loop is not unrolled or unrolled with no peel/tail loop.  
40% of peak computational performance is used (19.37 out of 48.00 FLOP per cycle (GFLOPS @ 1GHz))

gain potential hint expert

**Vectorization**

Your loop is fully vectorized, using full register length.

**Details**

All SSE/AVX instructions are used in vector version (process two or more data elements in vector registers).

**Execution units bottlenecks**

Performance is limited by execution of FP multiply or FMA (fused multiply-add) operations (the FP multiply/FMA unit is a bottleneck). By removing all these bottlenecks, you can lower the cost of an iteration from 19.00 to 12.50 cycles (1.52x speedup).

**Workaround**

Reduce the number of FP multiply/FMA instructions

# Valgrind Callgrind



## Callgrind : CPU emulation and function profiling KCacheGrind : profile results visualisation

The screenshot displays the Valgrind Callgrind interface. On the left, a table lists function calls with columns for 'Self', 'Called', 'Function', and 'Location'. The 'main' function is highlighted in blue. The right side shows a call graph visualization with nodes representing functions and edges representing calls. The graph is color-coded by function type.

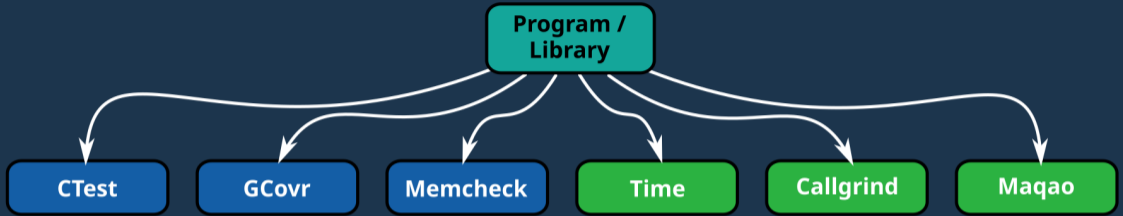
Self	Called	Function	Location
100.00	0.00	[0] int processAllFilePCon...	id-linux-x86-64.so.2
78.44	0.00	1   (below main)	phoenix_tex2html
100.00	0.00	1   lib_start_main@G...	libc.so.6:lib_start.c
100.00	0.00	1   (below main)	libc.so.6:lib_start_call_main.h
100.00	0.00	1   main	phoenix_tex2html:main.cpp:Option...
100.00	0.00	1   int processAllFilePCon...	phoenix_tex2html:main.cpp:basic_st...
78.44	0.00	1   PMultiFileParser:loadP...	(unknown)
78.44	0.00	1   PConfigParser:parseFile	libphoenix_file_parser.so.1.5.1:PMult...
78.44	0.00	562   <cycle 5>	(unknown)
78.44	0.00	562   PConfigParser:parseAll...	libphoenix_tex_2_html_lib.so.1.1.0:P...
78.42	0.96	15 989   <cycle 5>	(unknown)
55.41	0.26	320 473   PConfigParser:parseAll...	libphoenix_tex_2_html_lib.so.1.1.0:P...
45.43	0.00	2 226   parser_makingHighligh...	libphoenix_tex_2_html_lib.so.1.1.0:p...
45.43	0.01	2 226   parser_makingHighligh...	libphoenix_tex_2_html_lib.so.1.1.0:p...
42.05	0.00	258 319   parser_parseKeyword(P...	(unknown)
42.05	0.00	258 319   parser_parseKeyword(P...	libphoenix_tex_2_html_lib.so.1.1.0:p...
41.94	0.00	2 226   parser_parseKeyword(P...	(unknown)
41.94	0.91	7 246 264   parser_parseKeyword(P...	libphoenix_tex_2_html_lib.so.1.1.0:p...
31.42	4.94 140 600 557	PFfileParser:isMatchTok...	libphoenix_file_parser.so.1.5.1:PFfileP...
29.83	0.01	4 375 761   PFfileParser:isMatchTok...	(unknown)
29.82	1.24	4 375 761   PFfileParser:isMatchTok...	libphoenix_file_parser.so.1.5.1:PFfileP...
28.17	0.28 125 092 721	PFfileParser:isMatchTok...	(unknown)
26.41	0.00	310 953   PConfigParser:parsePa...	(unknown)
26.41	0.00	310 953   PConfigParser:parsePa...	libphoenix_tex_2_html_lib.so.1.1.0:P...
25.26	1.72 175 851 017	PFfileParser:isMatch(P...	libphoenix_file_parser.so.1.5.1:PFfileP...
24.79	0.38 169 944 094	PFfileParser:isMatch(P...	(unknown)
21.34	0.00	312 710   PConfigParser:parsePa...	(unknown)
21.34	0.02	312 710   PConfigParser:parsePa...	libphoenix_tex_2_html_lib.so.1.1.0:P...
19.88	0.00	1   plateobj_bookItem(P...	(unknown)
19.88	0.00	1   plateobj_bookItem(P...	libphoenix_tex_2_html_lib.so.1.1.0:pl...
19.87	0.00	1   plateobj_bookItem(P...	(unknown)
19.87	0.00	1   plateobj_bookItem(P...	libphoenix_tex_2_html_lib.so.1.1.0:pl...
19.87	0.00	1   plateobj_bookItem(P...	(unknown)
19.87	0.00	1   plateobj_bookItem(P...	libphoenix_tex_2_html_lib.so.1.1.0:pl...
19.84	0.00	88   <cycle 10>	(unknown)
18.59	0.38 169 793 896	PFfileParser:isMatch(P...	(unknown)
18.51	18.51 176 310 335	PFfileParser:isMatch(P...	libphoenix_file_parser.so.1.5.1:PFfileP...
13.41	0.00	486   plateobj_book_create...	libphoenix_tex_2_html_lib.so.1.1.0:pl...
11.38	0.00	1 263 468   PString:operator+PSt...	libphoenix_core.so.1.11.0:PString.cp...
11.38	0.00	1 263 985   PString:operator+PSt...	(unknown)
11.38	0.01	1 263 985   PString:add(PString co...	libphoenix_core.so.1.11.0:PString.cp...
11.37	0.00	1 264 502   PString:concatenateP...	(unknown)
11.37	0.07	1 264 502   PString:concatenateP...	libphoenix_core.so.1.11.0:PString.cp...
10.04	6.55 192 788 925	std::cxx11::basic_strin...	libstdc++.so.6.0.30
10.03	0.76 14 478 603	PString:concatenateP...	libphoenix_core.so.1.11.0:PString.cp...
9.82	0.03 14 203 510	PFfileParser:isMatch(P...	(unknown)
9.60	0.40 176 591 335	PFfileParser:isMatch(P...	libphoenix_file_parser.so.1.5.1:PFfileP...
9.55	0.75 16 620 388	void std::cxx11::basic...	libphoenix_core.so.1.11.0:basic_strin...
9.15	0.00	51 288   PFfileParser:isMatch(P...	(unknown)
9.15	0.00	51 288   PFfileParser:isMatch(P...	libphoenix_file_parser.so.1.5.1:PFfileP...

callgrind.out.70448 [1] Total (Instruction Fetch) Coût total : 88 294 722 485

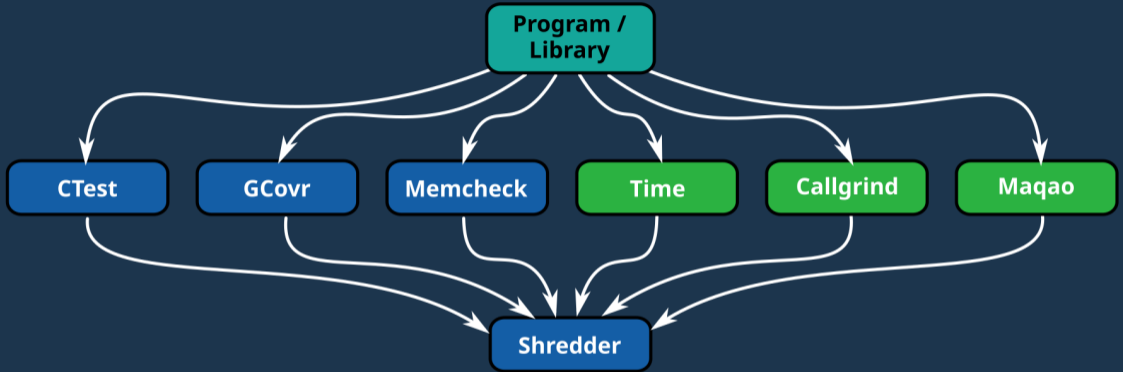


**Program /  
Library**

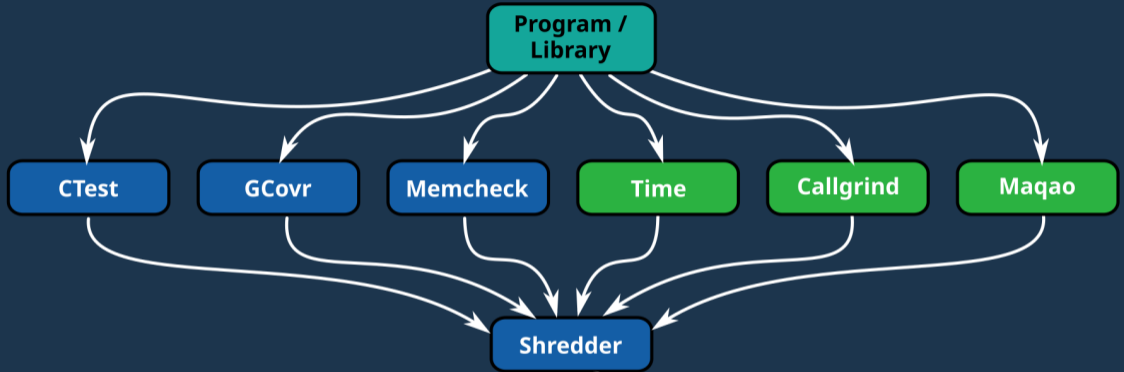
# Shredder



# Shredder

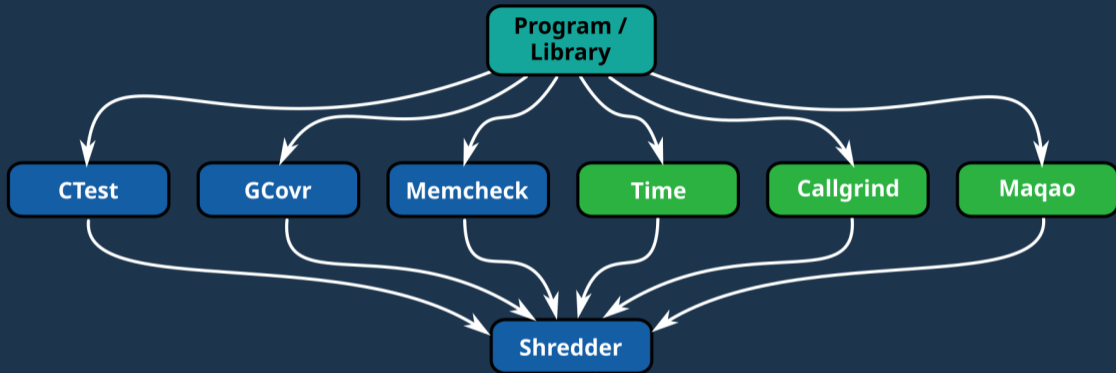


# Shredder



Gather **all tools**  
**report information**  
in a **single web site**

# Shredder



Packaged with **Pixi**

Gather **all tools**  
report information  
in a **single web site**





**Program /  
Library**

# Shredder as a service



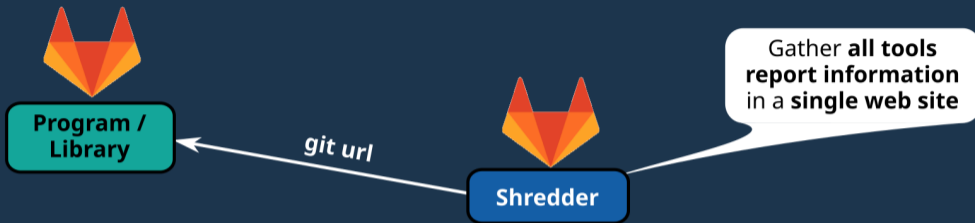
**Program /  
Library**



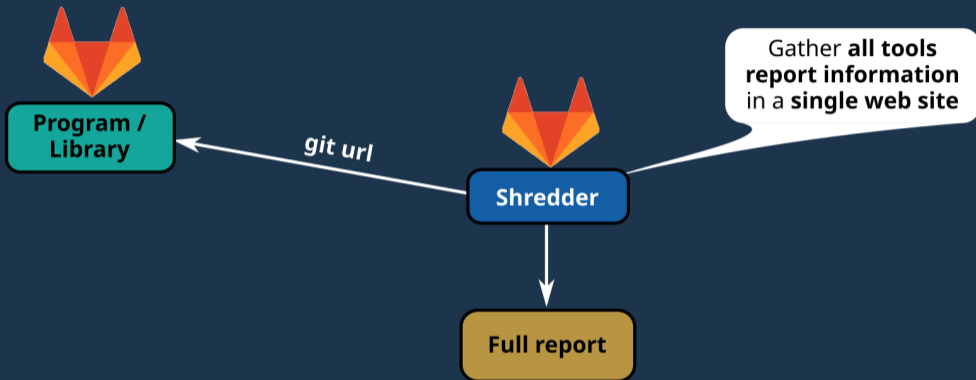
**Shredder**



# Shredder as a service



# Shredder as a service



**Shredder** gathers all report information from :

- **ctest**
- **gcovr**
- **valgrind** : **memcheck** / **callgrind**
- **maqao**
- **time**

- Missing **Valgrind Cadna** to check precision
- Missing **Malt**

