OPTIMIZING BUILDS
ON WINDOWS
SOME PRACTICAL CONSIDERATIONS

Alexandre Ganea, Ubisoft
alexandre.ganea@ubisoft.com

2019 Bay Area LLVM Developers' Meeting, Oct.22-23

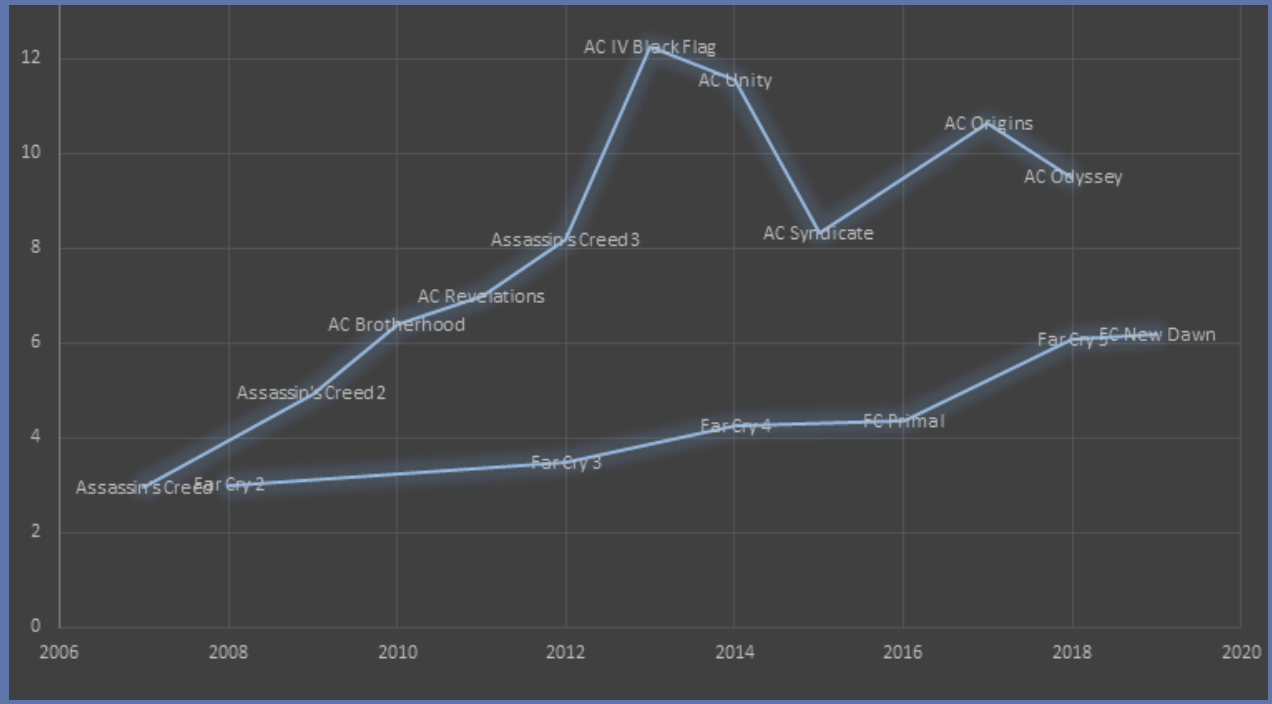
SUMMARY

PART 1

PREAMBLE:

CHALLENGES

Lines of Code
(Assassins' Creed, Far Cry)



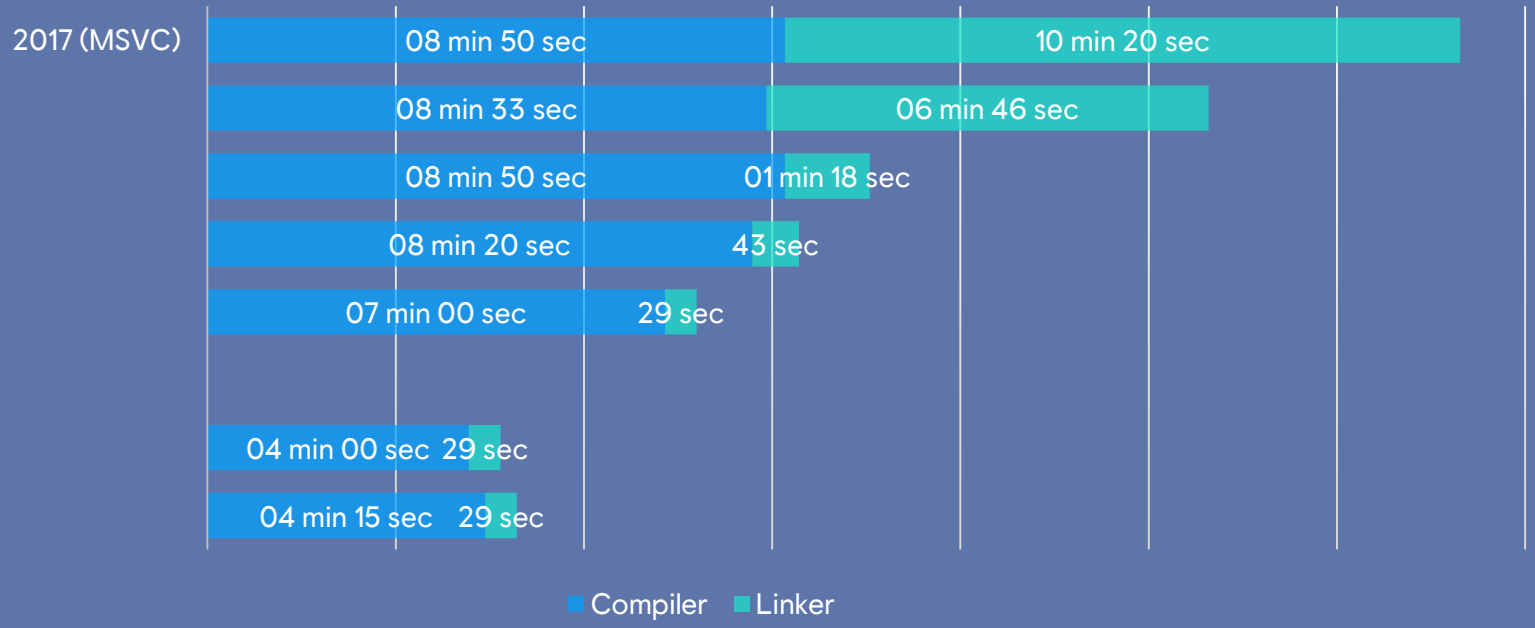
Game production constraints @ Ubisoft

Concurent AAA games	**20 – 25**
LoC/game	**30 - 50 M**
Programmers/title	**100 – 250**
Code Changes/day	**100 – 150 (peak:400)**
Build targets/platform	**5 – 6**
Platforms/Game	**4+**
Code workspace	**70 - 100 GB**
Data workspace	**100 - 200 GB**
Game builds/day	**100 – 150**
Stripped Build	**1 - 6 GB**
Final Build	**50 - 90 GB**

Editor Build	**20,000 .CPP**
	25,000 .H
	23 GB .OBJ
	9 GB .DEBUG$T
	10 M TYPE RECORDS
	42 M SYMBOLS
	300 M .EXE
	2 GB .PDB

Windows 10
Fastbuild, distributed
Always Unity builds

AAA GAME, CLEAN REBUILD
X64 EDITOR RELEASE (FASTBUILD)

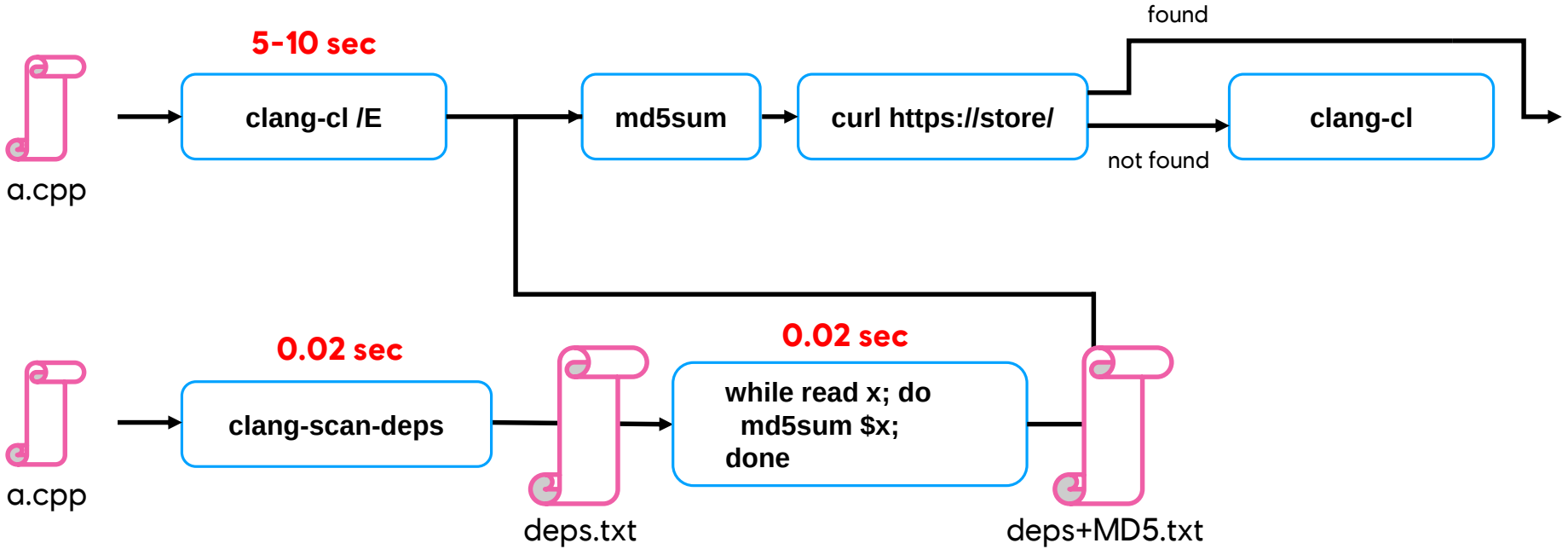


PART 2

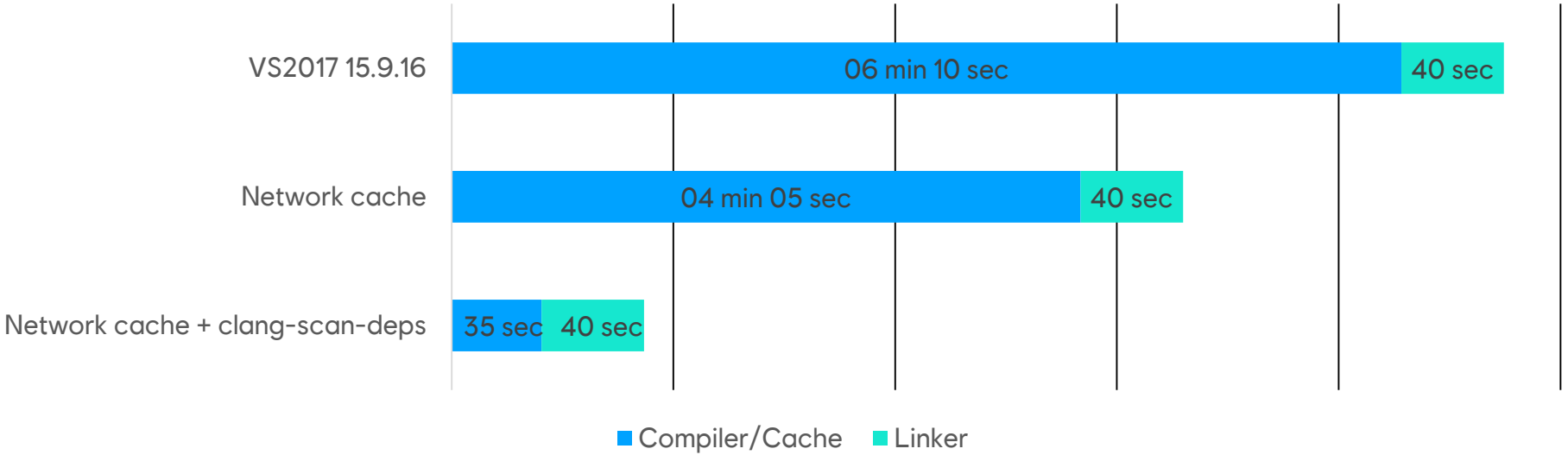
EXPERIMENTS

2.1
Clang-scan-deps & Fastbuild cache

FASTBUILD CACHE READ ALGORITHM

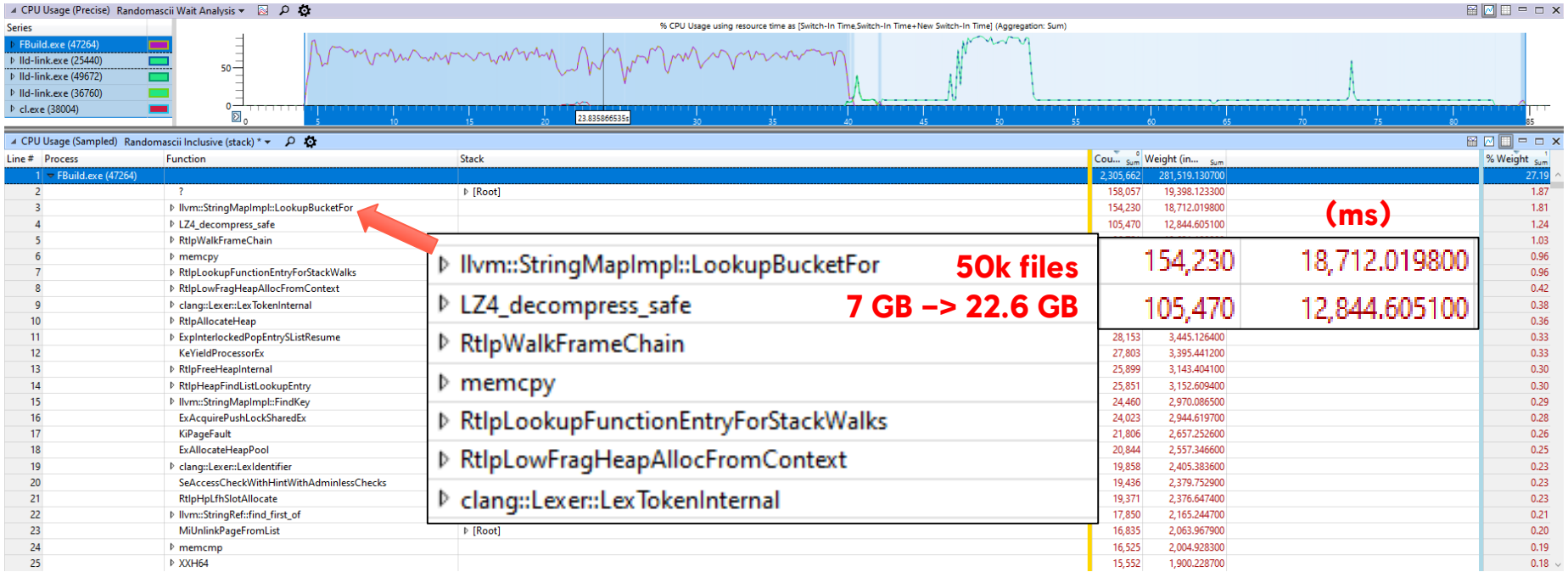


100% NETWORK CACHE HITS
AAA GAME, X64 EDITOR RELEASE (FASTBUILD)



clang-scan-deps
+ network cache

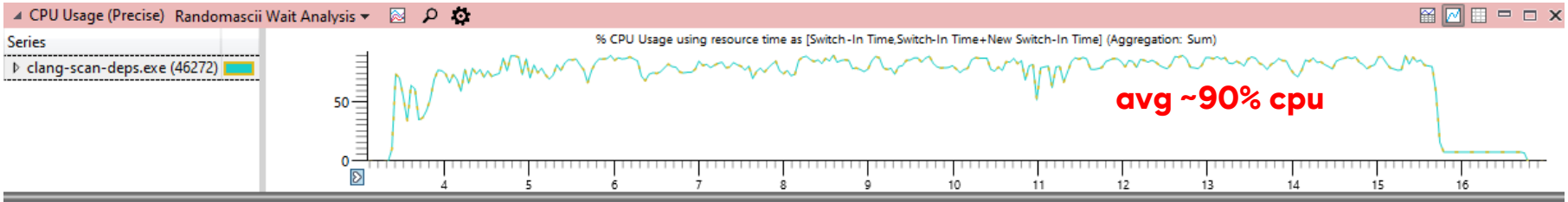
LLD
(MSVC OBJs + ghash)



Intel Xeon W-2135 @ 3.7 GHz, 128 GB, NVMe SSD, 1Gbps Network

2.2
StringMap

CLANG-SCAN-DEPS STANDALONE (50K FILES)



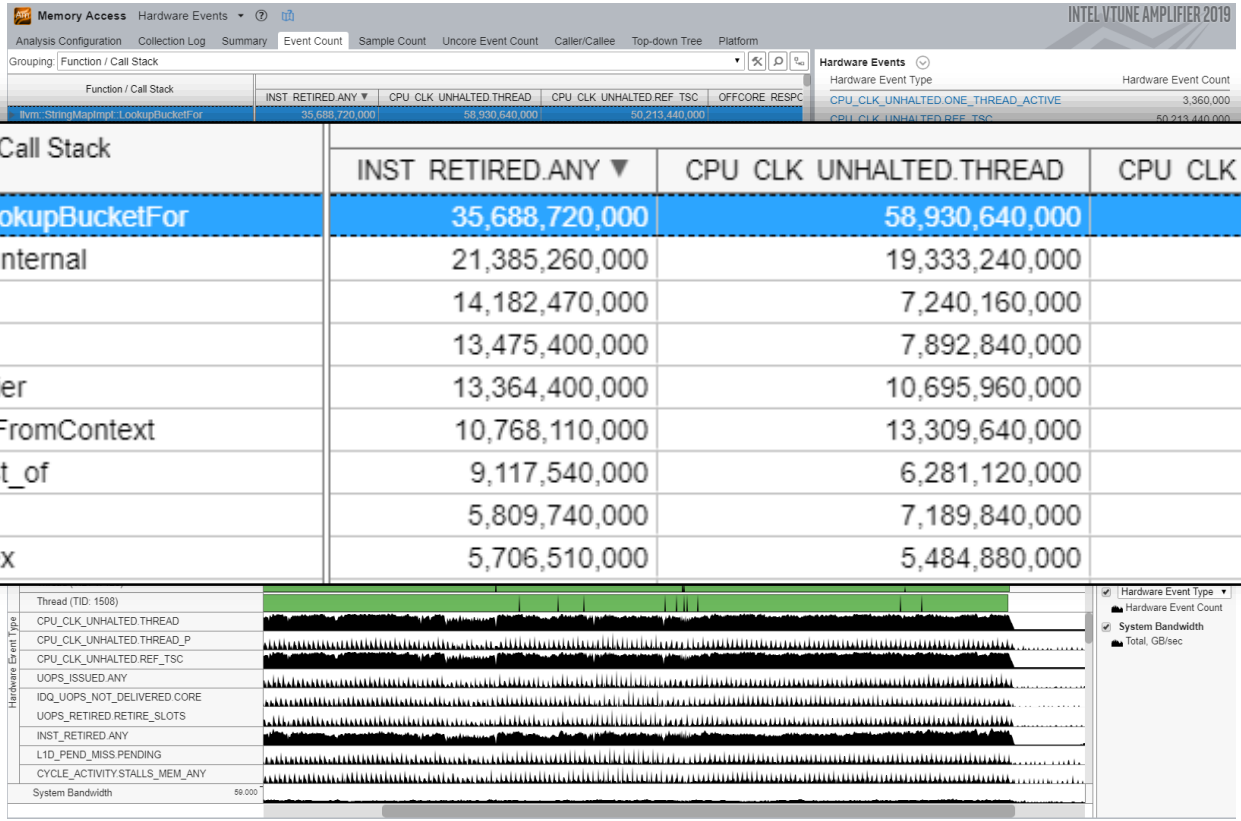
CPU Usage (Sampled) Randomascii Inclusive (stack) *

Line #	Process	Function	Stack	Cou... Sum	Weight (in... Sum	% Weig... Sum
1	clang-scan-deps.exe (46272)			978,995	119,435.575900	71.61
2		llvm::StringMapImpl::LookupBucketFor		113,661	13,832.426900	8.29
3		clang::Lexer::LexTokenInternal		31,391	3,815.536200	2.29
4		RtlpLowFragHeapAllocFromContext		25,158	3,066.277500	1.84
5		RtlpWalkFrameChain		20,612	2,521.816000	1.51
6		RtlpLookupFunctionEntryForStackWalks		19,812	2,424.523600	1.45
7		clang::Lexer::LexIdentifier		19,526	2,372.914000	1.42
8		RtlpFreeHeapInternal		18,493	2,251.169700	1.35
9		NtfsFindPrefixHashEntry	[Root]	17,938	2,194.988500	1.32
10		?	[Root]	16,981	2,061.071800	1.24
11		ExAcquirePushLockSharedEx	[Root]	16,102	1,968.971700	1.18
12		memcmp		14,980	1,822.660900	1.09
13		skipLine		14,887	1,823.887600	1.09

11.5% process time

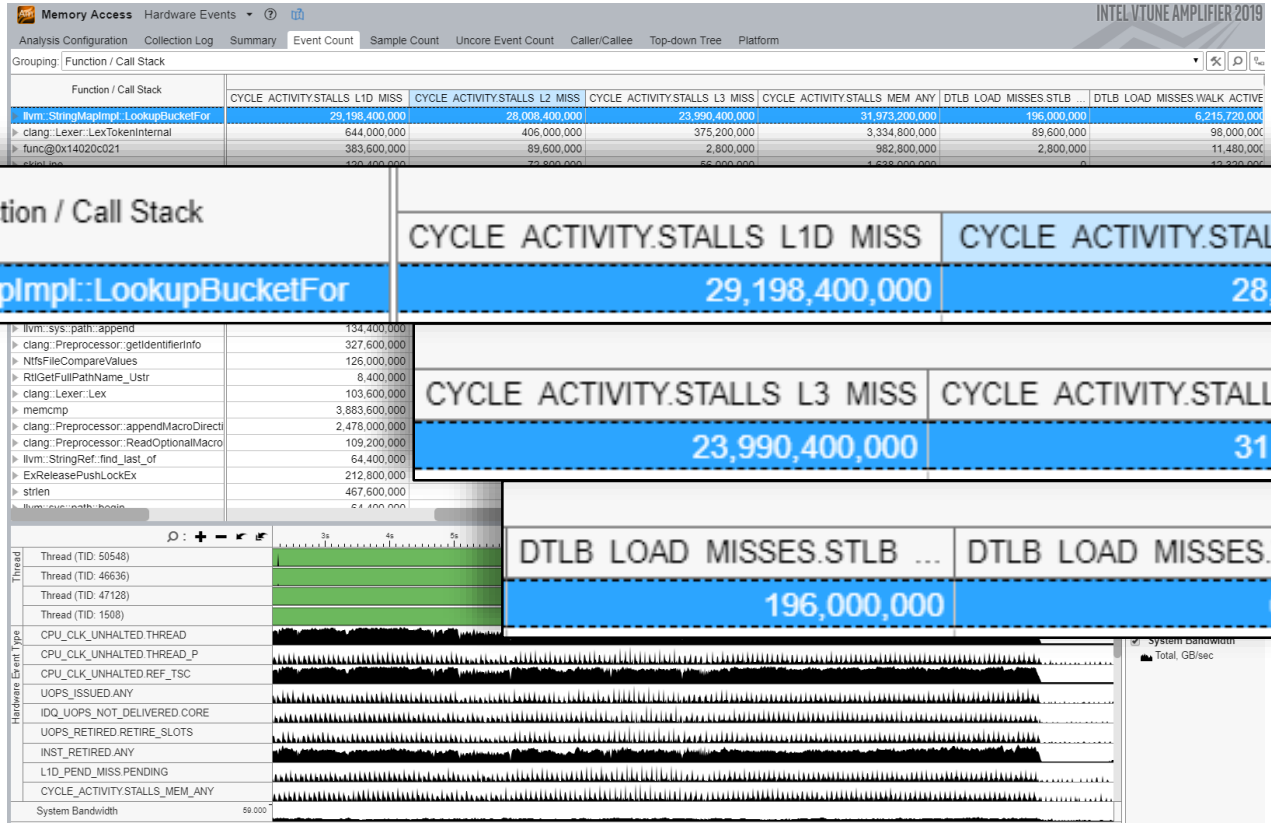
Intel Xeon W-2135 @ 3.7 GHz (6-core), 128 GB, NVMe SSD

STRINGMAP



Function / Call Stack	INST_RETIRED.ANY ▼	CPU_CLK_UNHALTED.THREAD	CPU_CLK_UNHALTED.REF_TSC
llvm::StringMapImpl::LookupBucketFor	35,688,720,000	58,930,640,000	50,213,440,000
clang::Lexer::LexTokenInternal	21,385,260,000	19,333,240,000	16,342,160,000
func@0x14020c021	14,182,470,000	7,240,160,000	6,174,560,000
skipLine	13,475,400,000	7,892,840,000	6,498,680,000
clang::Lexer::LexIdentifier	13,364,400,000	10,695,960,000	8,891,840,000
RtlpLowFragHeapAllocFromContext	10,768,110,000	13,309,640,000	11,129,600,000
llvm::StringRef::find_first_of	9,117,540,000	6,281,120,000	5,298,400,000
memcpy	5,809,740,000	7,189,840,000	6,048,760,000
clang::Preprocessor::Lex	5,706,510,000	5,484,880,000	4,651,640,000

STRINGMAP


Memory Access
Hardware Events
INTEL VTUNE AMPLIFIER 2019
Analysis Configuration
Collection Log
Summary
Event Count
Sample Count
Uncore Event Count
Caller/Callee
Top-down Tree
Platform
Grouping: Function / Call Stack
Function / Call Stack
CYCLE_ACTIVITY.STALLS_L1D_MISS
CYCLE_ACTIVITY.STALLS_L2_MISS
CYCLE_ACTIVITY.STALLS_L3_MISS
CYCLE_ACTIVITY.STALLS_MEM_ANY
DTLB_LOAD_MISSES.STLB_...
DTLB_LOAD_MISSES.WALK_ACTIVE
llvm::StringMapImpl::LookupBucketFor 29,198,400,000 28,008,400,000 23,990,400,000 31,973,200,000 196,000,000 6,215,720,000
clang::Lexer::LexTokenInternal 644,000,000 406,000,000 375,200,000 3,334,800,000 89,600,000 98,000,000
func@0x14020c021 383,600,000 89,600,000 2,800,000 982,800,000 2,800,000 11,480,000
llvm::sys::path::append 134,400,000
clang::Preprocessor::getIdentifierInfo 327,600,000
NtfsFileCompareValues 126,000,000
RtlGetFullPathName_Ustr 8,400,000
clang::Lexer::Lex 103,600,000
memcmp 3,883,600,000
clang::Preprocessor::appendMacroDirecti 2,478,000,000
clang::Preprocessor::ReadOptionalMacro 109,200,000
llvm::StringRef::find_last_of 64,400,000
ExReleasePushLockEx 212,800,000
strlen 467,600,000
Thread
Thread (TID: 50548)
Thread (TID: 46636)
Thread (TID: 47128)
Thread (TID: 1508)
Hardware Event Type
CPU_CLK_UNHALTED.THREAD
CPU_CLK_UNHALTED.THREAD_P
CPU_CLK_UNHALTED.REF_TSC
UOPS_ISSUED.ANY
IDQ_UOPS_NOT_DELIVERED.CORE
UOPS_RETIRED.RETIRE_SLOTS
INST_RETIRED.ANY
L1D_PEND_MISS.PENDING
CYCLE_ACTIVITY.STALLS_MEM_ANY
System Bandwidth
59.000
System Bandwidth
Total, GB/sec

DOWN THE RABBIT HOLE

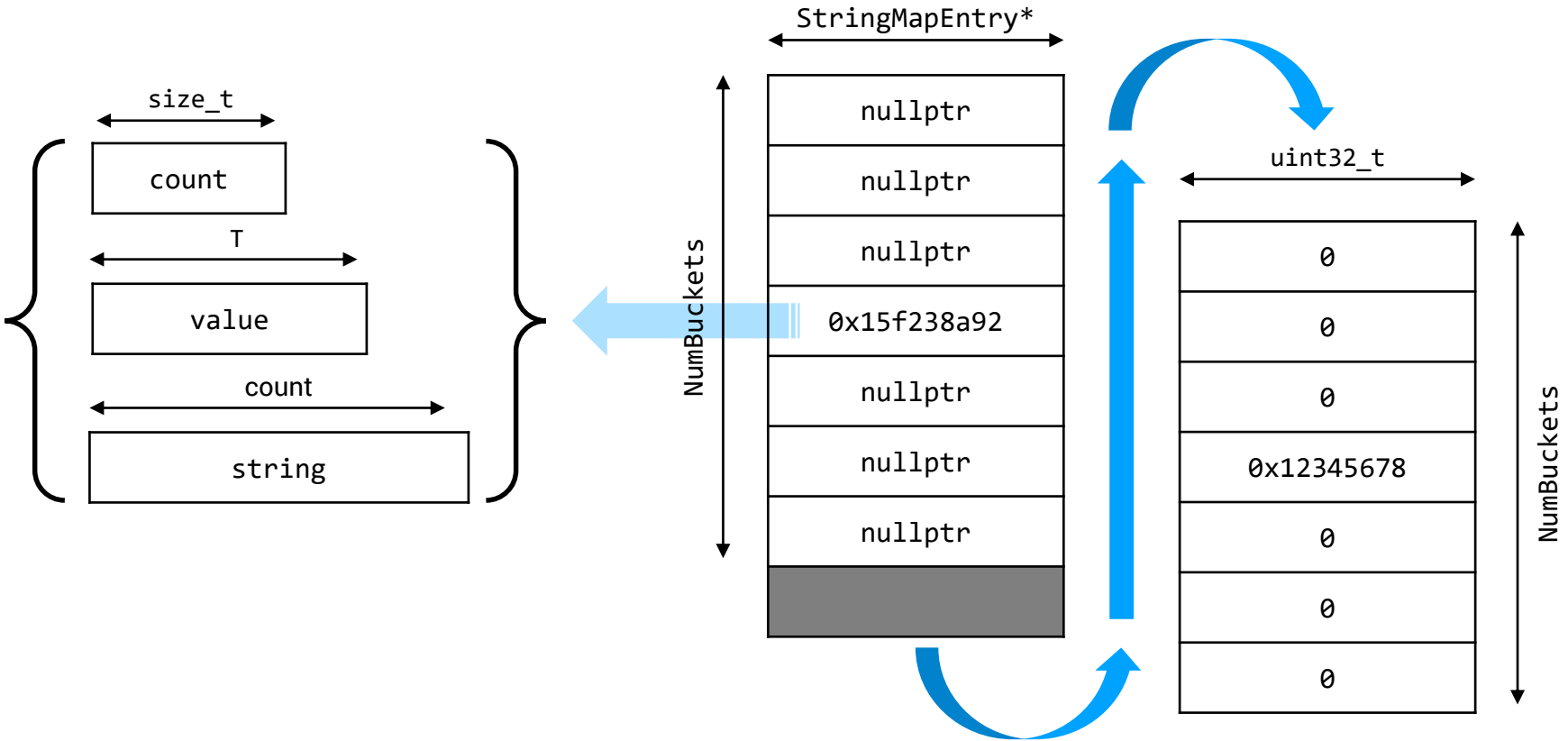
```
sizeof(std::error_code) -> 16 bytes

sizeof(llvm::ErrorOr<DirectoryEntry&>) -> 24 bytes

sizeof(llvm::StringMapEntry<llvm::ErrorOr<DirectoryEntry&>>) –> 32 bytes (+string contents)
```

llvm::StringMapEntry<llvm::ErrorOr<clang::DirectoryEntry > >

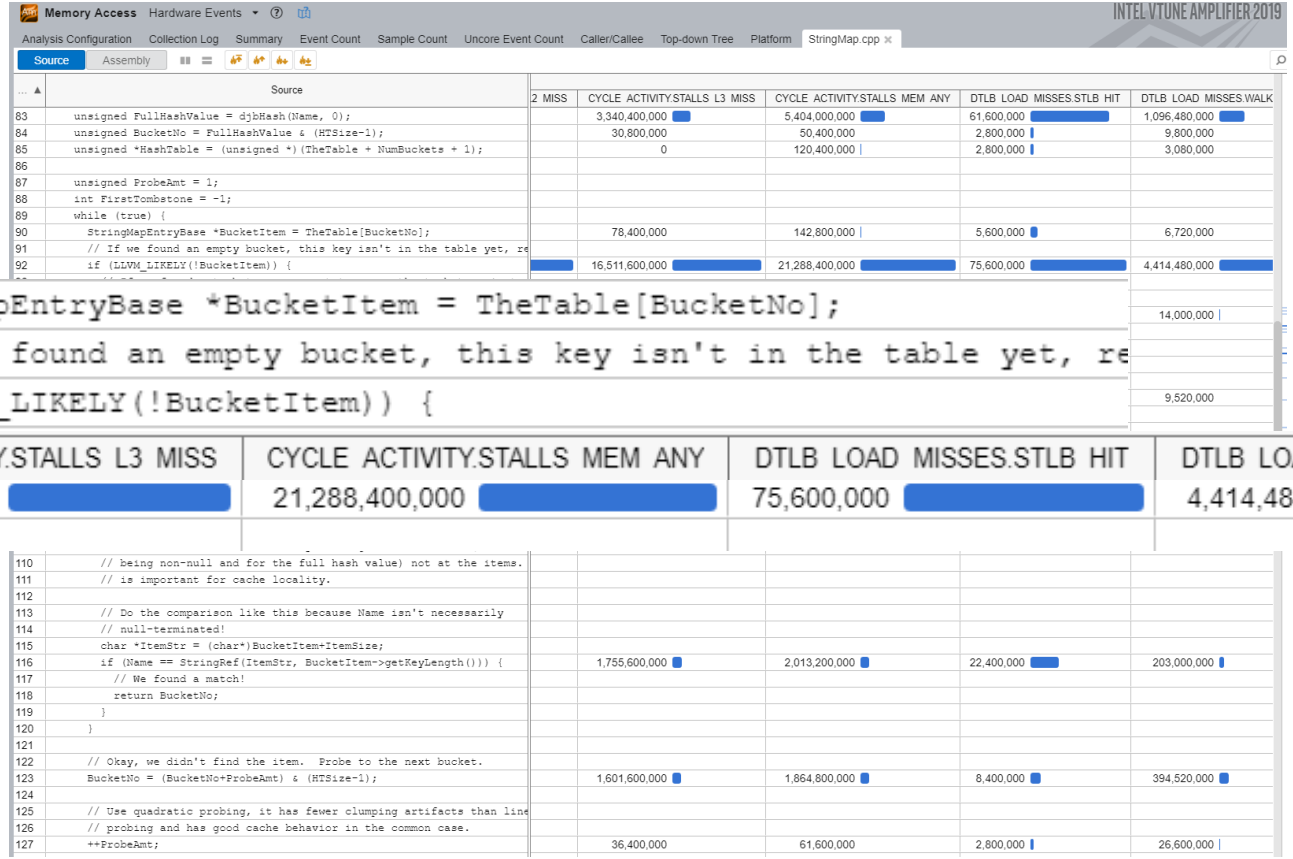
Members | Functions

	Field	Type	Offset	Bit offset	Size	Padding
-	Base: llvm::StringMapEntryBase	llvm::StringMapEntryBase	0		8 bytes	0
	|__ StrLen	uint64	0		8 bytes	
-	second	llvm::ErrorOr<clang::DirectoryEntry &>	8		24 bytes	7
+	|__ ErrorStorage	llvm::AlignedCharArrayUnion<std::e...	8		16 bytes	0
+	|__ TStorage	llvm::AlignedCharArrayUnion<std::r...	8		8 bytes	0
	|__ HasError	bool	24	0	1 bit	
		Padding	17	7	7	

STRINGMAP: MEMORY LAYOUT



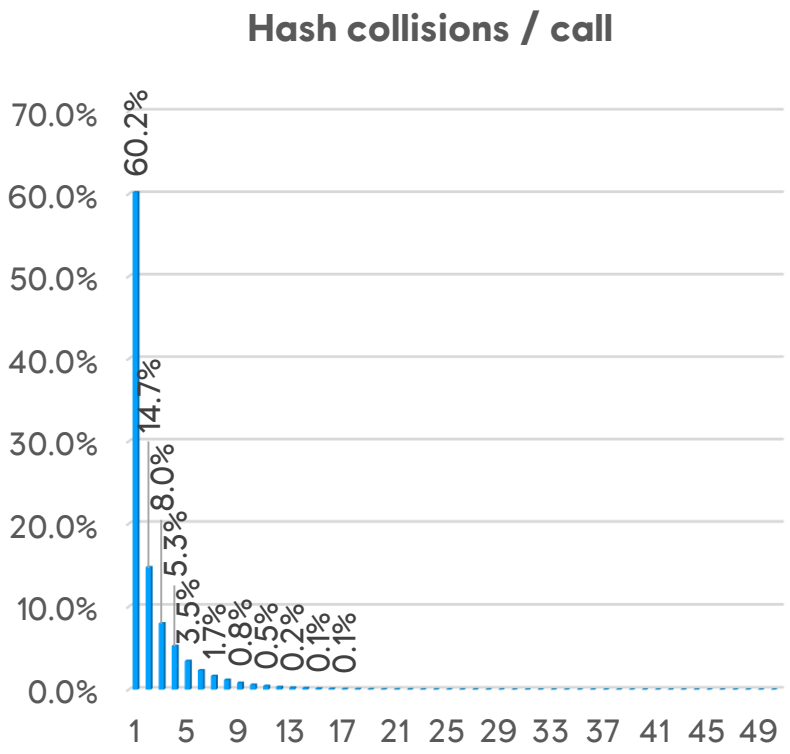
STRINGMAP (VTUNE)

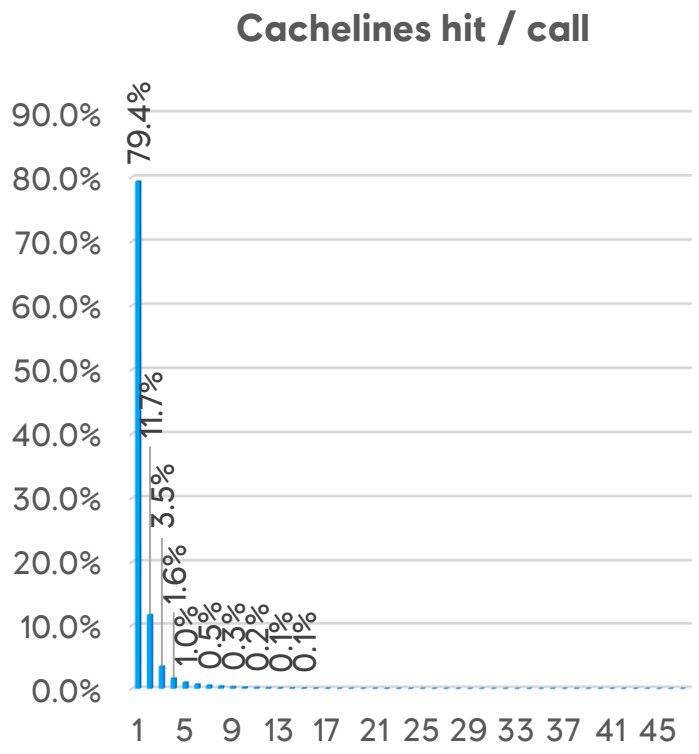



```
StringMapEntryBase *BucketItem = TheTable[BucketNo];
// If we found an empty bucket, this key isn't in the table yet, re
if (LLVM_LIKELY(!BucketItem)) {
```

CYCLE_ACTIVITY.STALLS_L3_MISS	CYCLE_ACTIVITY.STALLS_MEM_ANY	DTLB_LOAD_MISSES.STLB_HIT	DTLB_LOAD_MISSES.WALK
16,511,600,000	21,288,400,000	75,600,000	4,414,480,000

STRINGMAP STATS





187 M samples

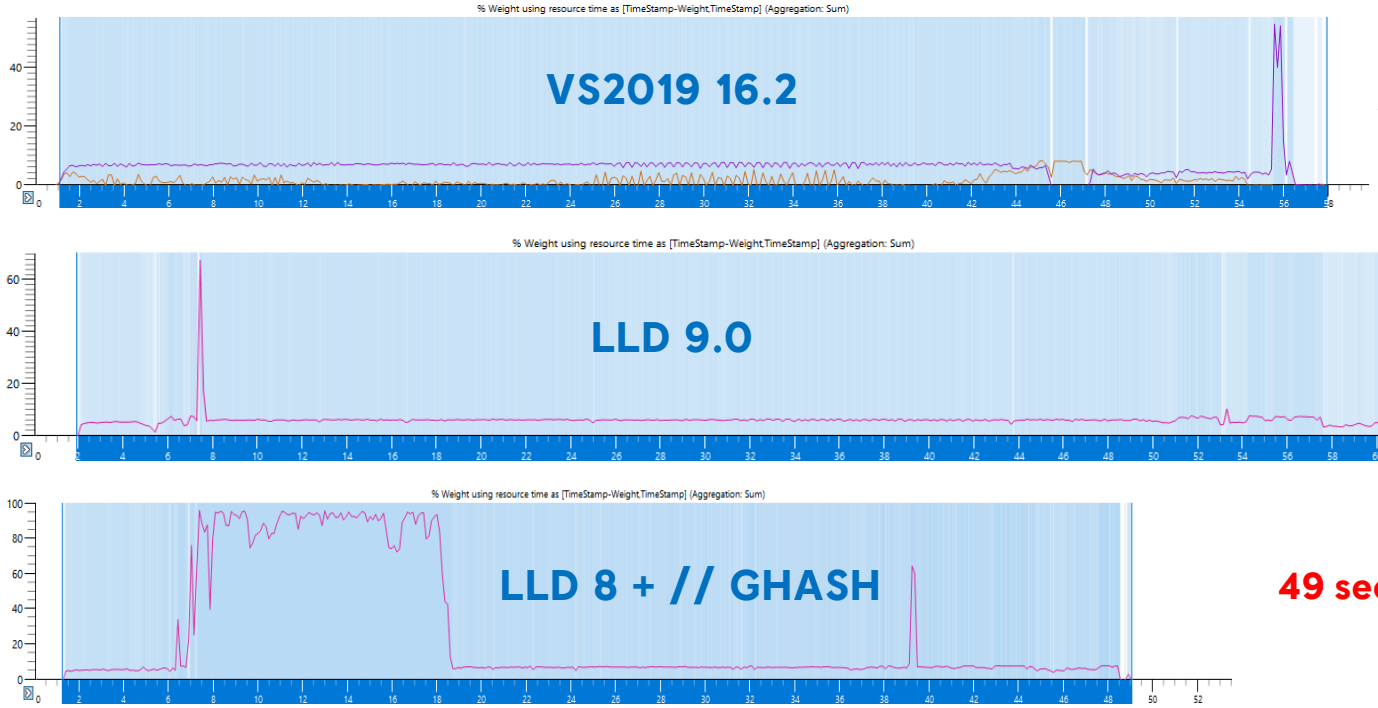
DenseMap<uint64_t,T> + xxHash64()
+ StringSaver

DenseMap<__int128,T> + XXH128()
+ StringSaver

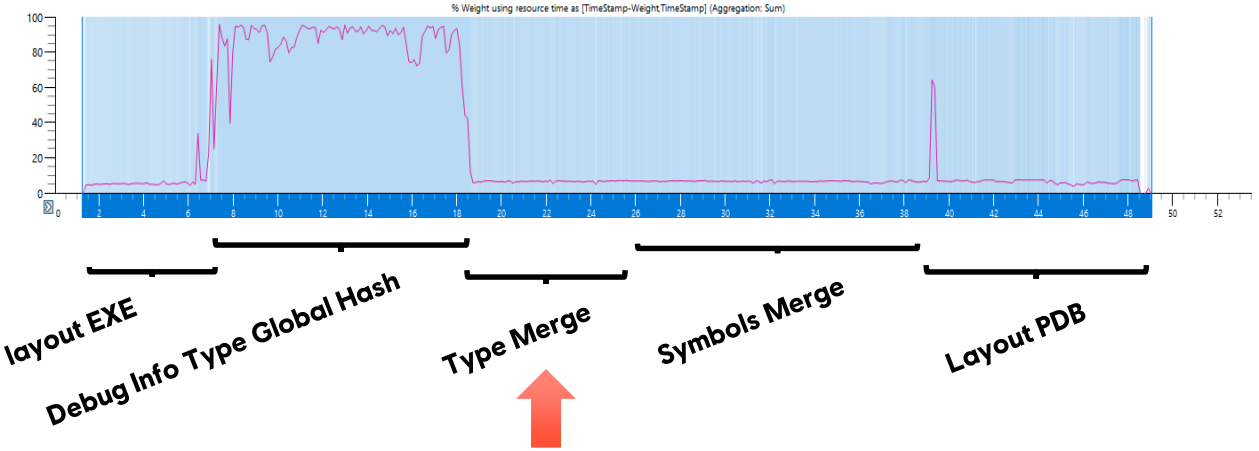
2.4
Multithreading LLD (COFF driver)

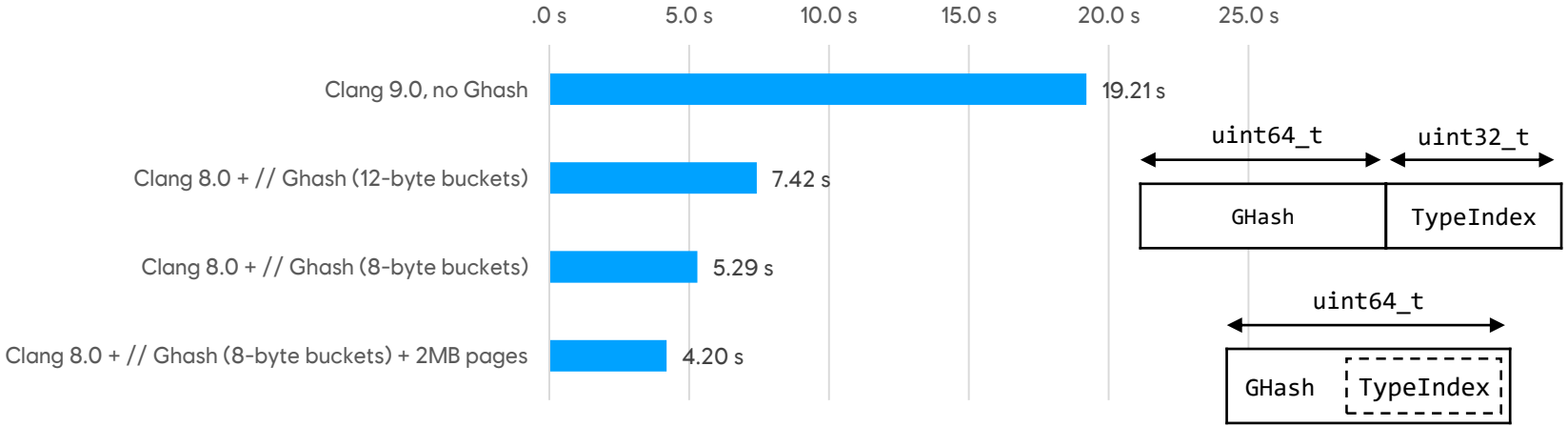
LINK AAA GAME, X64 EDITOR RELEASE

(22.8GB MSVC OBJS)



Intel Xeon W-2135 @ 3.7 GHz (6-core), 128 GB, NVMe SSD


% Weight using resource time as [TimeStamp-Weight,TimeStamp] (Aggregation: Sum)
Read sections & layout EXE
Debug Info Type Global Hash
Type Merge
Symbols Merge
Layout PDB

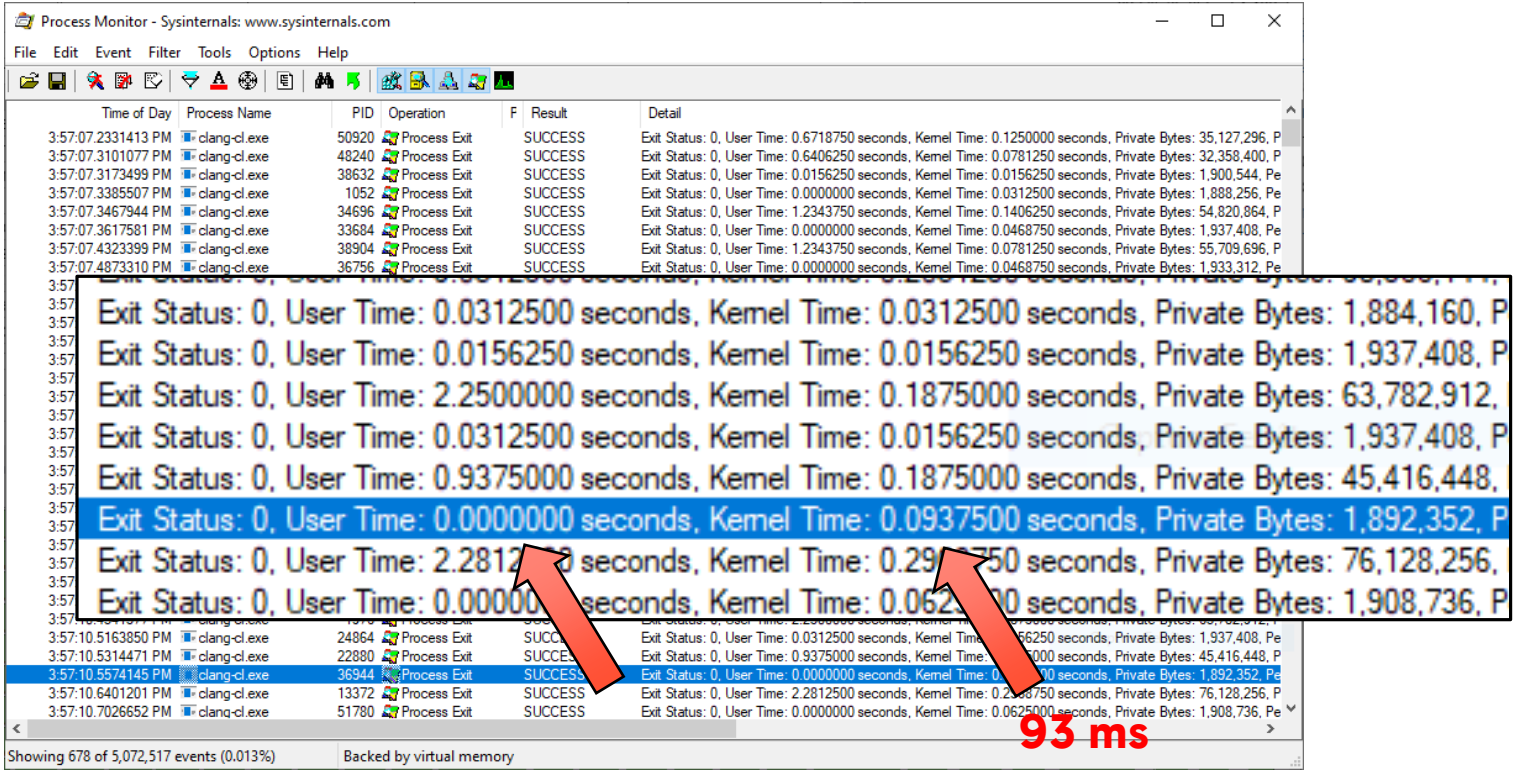

.0 s
5.0 s
10.0 s
15.0 s
20.0 s
25.0 s
Clang 9.0, no Ghash
19.21 s
Clang 8.0 + // Ghash (12-byte buckets)
7.42 s
Clang 8.0 + // Ghash (8-byte buckets)
5.29 s
Clang 8.0 + // Ghash (8-byte buckets) + 2MB pages
4.20 s
uint64_t
uint32_t
GHash
TypeIndex
uint64_t
GHash
TypeIndex

2.3
Process Creation

COMPILING WITH CLANG 9.0

ninja.exe	0.29	30,332 K	31,500 K	4,276,340 K	23620
clang-cl.exe		2,180 K	10,248 K	4,355,108 K	28944
clang-cl.exe	8.14	275,144 K	300,784 K	4,640,804 K	31480
clang-cl.exe		2,192 K	10,252 K	4,355,108 K	7564
clang-cl.exe	8.25	256,096 K	281,312 K	4,622,340 K	26956
clang-cl.exe		2,208 K	10,272 K	4,355,108 K	8064
clang-cl.exe	7.46	167,672 K	195,076 K	4,562,456 K	35484
clang-cl.exe		2,200 K	10,264 K	4,355,108 K	14292
clang-cl.exe	8.10	143,192 K	170,672 K	4,513,596 K	23720
clang-cl.exe		2,160 K	10,220 K	4,355,108 K	14232
clang-cl.exe	8.70	167,492 K	186,804 K	4,532,180 K	7480
clang-cl.exe		2,164 K	10,224 K	4,355,108 K	30776
clang-cl.exe	6.45	144,060 K	161,972 K	4,516,752 K	20112
clang-cl.exe		2,160 K	10,224 K	4,355,108 K	29408
clang-cl.exe	7.21	102,432 K	124,076 K	4,464,760 K	15044
clang-cl.exe		2,260 K	10,296 K	4,355,108 K	16404
clang-cl.exe	8.40	101,684 K	122,984 K	4,463,836 K	19840
clang-cl.exe		2,248 K	10,288 K	4,364,900 K	31124
clang-cl.exe	7.14	91,104 K	108,872 K	4,465,128 K	17324
clang-cl.exe		2,156 K	10,220 K	4,355,108 K	25252
clang-cl.exe	6.09	86,904 K	103,776 K	4,451,788 K	15116
clang-cl.exe		2,156 K	10,220 K	4,355,108 K	26800
clang-cl.exe	6.30	69,224 K	86,408 K	4,430,528 K	34252
clang-cl.exe		2,148 K	10,212 K	4,355,108 K	34484
clang-cl.exe	5.29	66,224 K	83,432 K	4,430,780 K	24736
clang-cl.exe		2,156 K	10,220 K	4,355,108 K	30684
clang-cl.exe	5.91	58,804 K	75,688 K	4,430,440 K	26188
clang-cl.exe		2,160 K	10,228 K	4,355,108 K	27188
clang-cl.exe	6.64	45,792 K	63,144 K	4,413,180 K	29616
clang-cl.exe	0.27	2,216 K	10,272 K	4,355,108 K	32748
clang-cl.exe	2.30	17,004 K	32,716 K	4,377,492 K	32024
clang-cl.exe	0.27	2,160 K	10,224 K	4,355,108 K	25944
clang-cl.exe	1.58	12,936 K	27,728 K	4,386,724 K	34904
clang-cl.exe	0.29	2,172 K	10,232 K	4,355,108 K	15304
clang-cl.exe	< 0.01	936 K	108 K	88,984 K	19824

CLANG CC1 IN PROCMON



MAKING CC1 REENTRANT

clang/tools/driver/driver.cpp

```
int main(int argc_, const char **argv_) {
  noteBottomOfStack();
  llvm::InitLLVM X(argc_, argv_);
  SmallVector<const char *, 256> argv(argv_, argv_ + argc_);

  if (llvm::sys::Process::FixupStandardFileDescriptors())
    return 1;

  llvm::InitializeAllTargets();
  return ClangDriverMain(argv);
}

int ClangDriverMain(SmallVectorImpl<const char *>& argv) {
  static LLVM_THREAD_LOCAL bool EnterPE = true;
  if (EnterPE) {
    llvm::sys::DynamicLibrary::AddSymbol("ClangDriverMain", (void*)(i..
    EnterPE = false;
  } else {
    llvm::cl::ResetAllOptionOccurrences();
  }

  auto TargetAndMode = ToolChain::getTargetAndModeFromProgramName(arg..
```

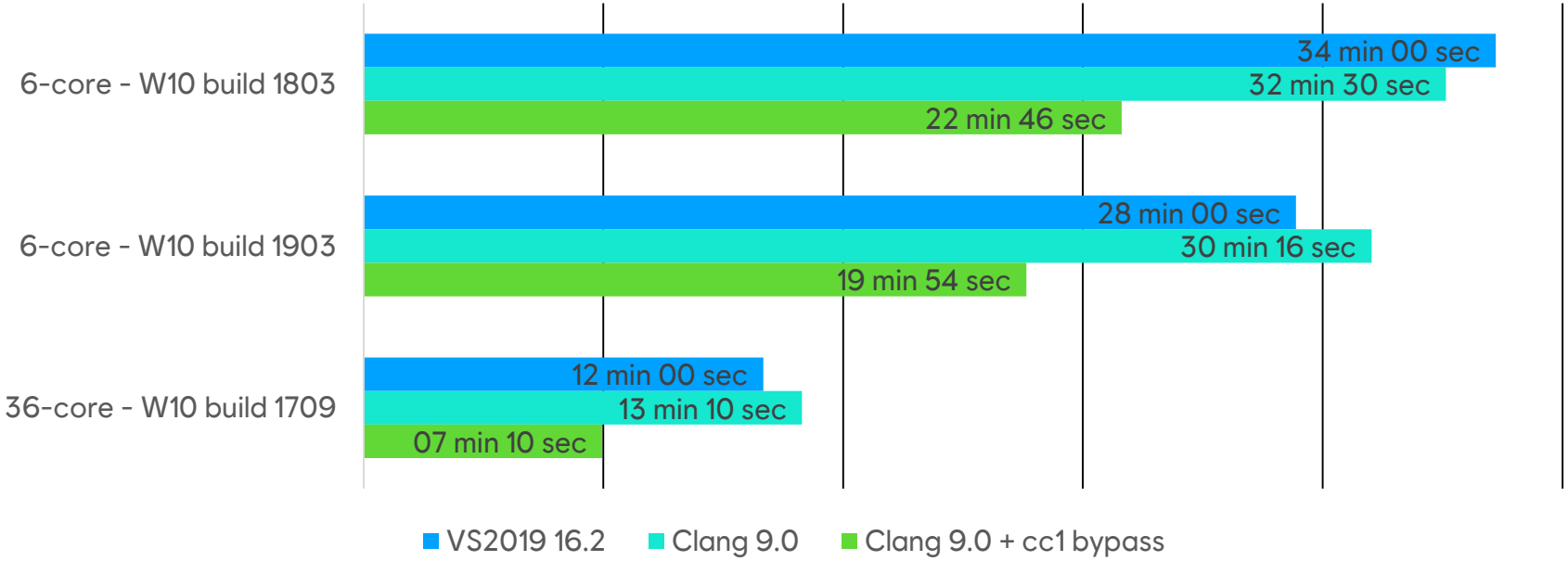
clang/lib/driver/Job.cpp

```
int Command::Execute(ArrayRef<llvm::Optional<StringRef>> Redirects,
                     std::string *ErrMsg, bool *ExecutionFailed) const {
  [...]
  typedef int (*ClangDriverMainFunc)(SmallVectorImpl<const char *> &);
  ClangDriverMainFunc ClangDriverMain = nullptr;

  [...]
  if (ClangDriverMain) {
    [...]
    llvm::CrashRecoveryContext CRC;
    CRC.EnableExceptionHandler = true;

    const void *PrettyState = llvm::SavePrettyStackState();

    int Ret = 0;
    auto ExecuteClangMain = [&]() { Ret = ClangDriverMain(Argv); };

    if (!CRC.RunSafely(ExecuteClangMain)) {
      llvm::RestorePrettyStackState(PrettyState);
      return CRC.RetCode;
    }
    return Ret;
  } else {
    auto Args = llvm::toStringRefArray(Argv.data());
    return llvm::sys::ExecuteAndWait(Executable, Args, Env, Redirects,
                                     /*secondsToWait*/ 0,
                                     /*memoryLimit*/ 0, ErrMsg,
                                     ExecutionFailed);
  }
}
```

CLANG DRIVER & CC1 MERGED

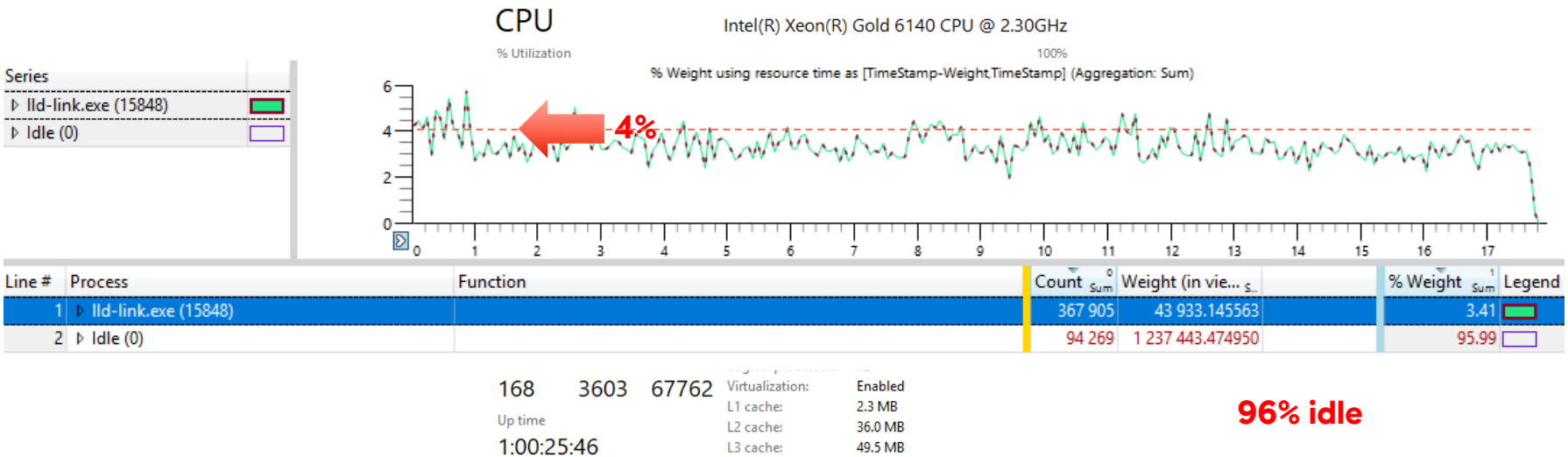
ninja.exe	0.25	101,580 K	101,348 K	4,356,428 K	8220
clang-cl.exe	8.79	261,936 K	283,840 K	4,638,060 K	51044
clang-cl.exe	8.70	341,664 K	364,680 K	4,702,980 K	12980
clang-cl.exe	8.21	225,288 K	251,888 K	4,599,624 K	28236
clang-cl.exe	6.31	201,144 K	227,548 K	4,564,128 K	7480
clang-cl.exe	5.89	273,580 K	289,808 K	4,671,332 K	50408
clang-cl.exe	6.83	176,852 K	203,424 K	4,548,028 K	17296
clang-cl.exe	8.74	165,520 K	186,028 K	4,532,868 K	48632
clang-cl.exe	8.87	173,024 K	192,700 K	4,547,932 K	7276
clang-cl.exe	7.61	116,856 K	139,108 K	4,479,316 K	49772
clang-cl.exe	6.42	108,768 K	130,916 K	4,479,288 K	30200
clang-cl.exe	5.40	75,856 K	96,612 K	4,448,360 K	33116
clang-cl.exe	6.02	64,732 K	85,696 K	4,429,460 K	46696
clang-cl.exe	5.70	52,180 K	70,856 K	4,423,088 K	24380
clang-cl.exe	3.99	33,452 K	51,036 K	4,395,836 K	21816
clang-cl.exe	1.47	12,748 K	27,664 K	4,378,196 K	48468
clang-cl.exe	1.77	14,000 K	29,160 K	4,378,236 K	26784
clang-cl.exe	2.56	19,448 K	35,604 K	4,378,832 K	52520

BYPASSING THE CC1 PROCESS
CLEAN REBUILD LLVM, CLANG & LLD



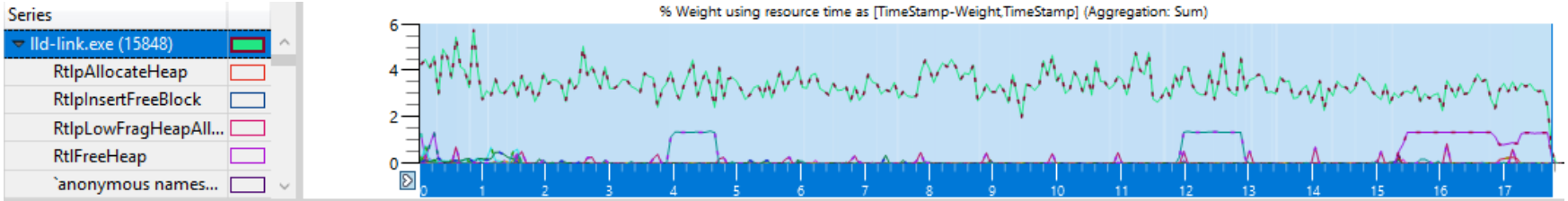
2.5
CRT Allocator

LINKING RAINBOW6: SIEGE WITH THINLTO :-(



96% idle

THINLTO: ALLOCATOR CONTENTION



Line #	Process	Function	Count Sum 0	Weight (in vie... s	% Weight Sum 1	Legend
1	lld-link.exe (15848)		367 905	43 933.145563	3.41	
2		RtlpAllocateHeap	80 788	9 410.376642	0.73	
3		RtlpInsertFreeBlock	48 334	5 654.658752	0.44	
4		RtlpLowFragHeapAllocFromContext	13 004	1 567.064625	0.12	
5		RtlFreeHeap	6 268	758.850566	0.06	
6		`anonymous namespace'::Verifier::visitMDNode	4 383	533.541286	0.04	
7		llvm::FunctionImportGlobalProcessing::processGlobalForThinLTO	4 009	489.450649	0.04	
8		llvm::SimpleBitstreamCursor::Read	3 787	457.680578	0.04	
9		llvm::SmallPtrSetImplBase::insert_imp_big	3 763	457.452098	0.04	
10		llvm::Use::getUser	3 668	444.645922	0.03	
11		llvm::Value::getName	2 832	344.016738	0.03	
12		llvm::ValueHandleBase::AddToUseList	2 729	330.571043	0.03	
13		llvm::StringMapImpl::LookupBucketFor	2 659	321.547708	0.02	
14		llvm::SmallPtrSetImpl<llvm::cl::SubCommand *>::insert	2 376	288.285001	0.02	

REPLACING THE CRT ALLOCATOR

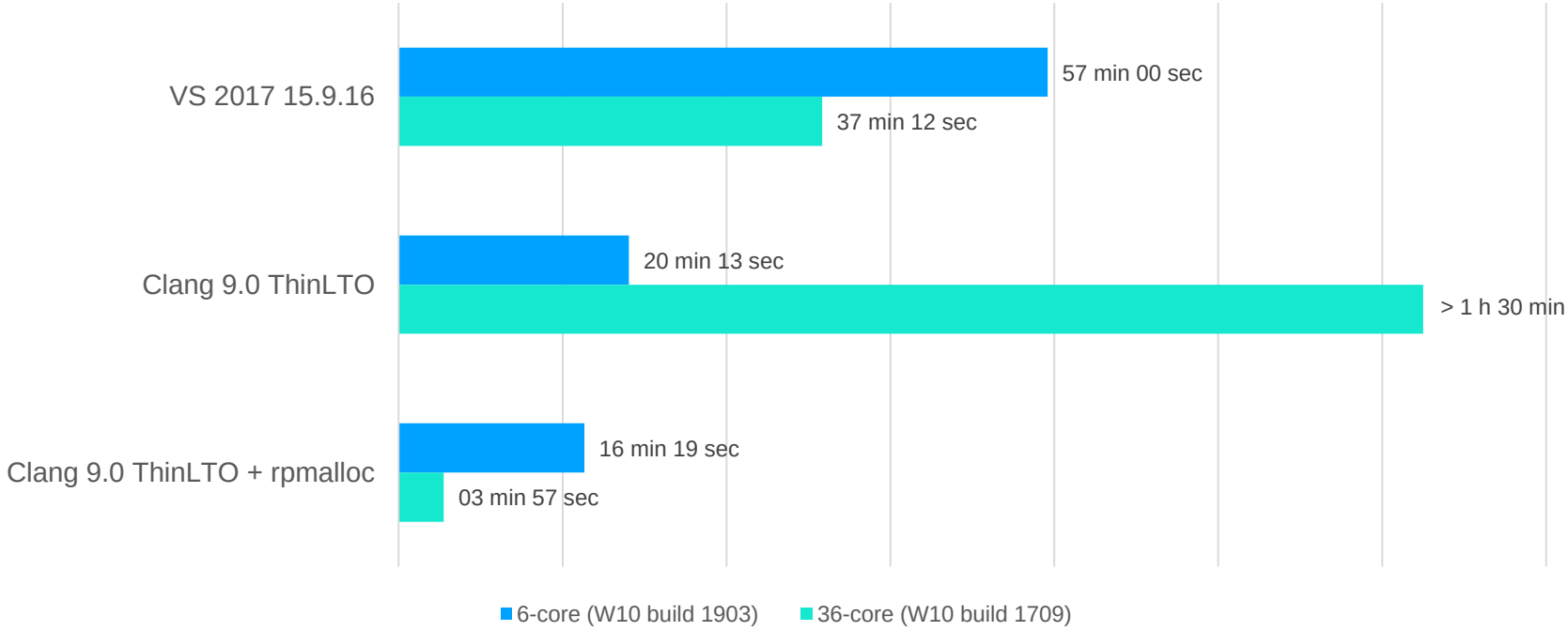
llvm/lib/Support/Windows/Memory.inc

```
#include "rpmalloc/rpmalloc.c"

extern "C" {
_ACRTIMP _CRTRESTRICT void *malloc(size_t size) {
  return rpmalloc(size);
}

_ACRTIMP void free(void *p) { rpfree(p); }

_ACRTIMP _CRTRESTRICT void *calloc(size_t n, size_t elem_size) {
  return rpcalloc(n, elem_size);
}
_ACRTIMP _CRTRESTRICT void *realloc(void *ptr, size_t size) {
  return rprealloc(ptr, size);
}
}

// Bypass CRT debug allocator
#ifdef _DEBUG
void *operator new(decltype(sizeof(0)) n) noexcept(false) { return malloc(n); }
void __CRTDECL operator delete(void *const block) noexcept { free(block); }

void *operator new[](std::size_t s) throw(std::bad_alloc) { return malloc(s); }
void operator delete[](void *p) throw() { free(p); }
#endif
```

$LD_PRELOAD=/path/to/my/malloc.so /bin/ls

https://github.com/mjansson/rpmalloc

THINLTO (CLEAN REBUILD)
RAINBOW 6: SIEGE, PC GAME PROFILE

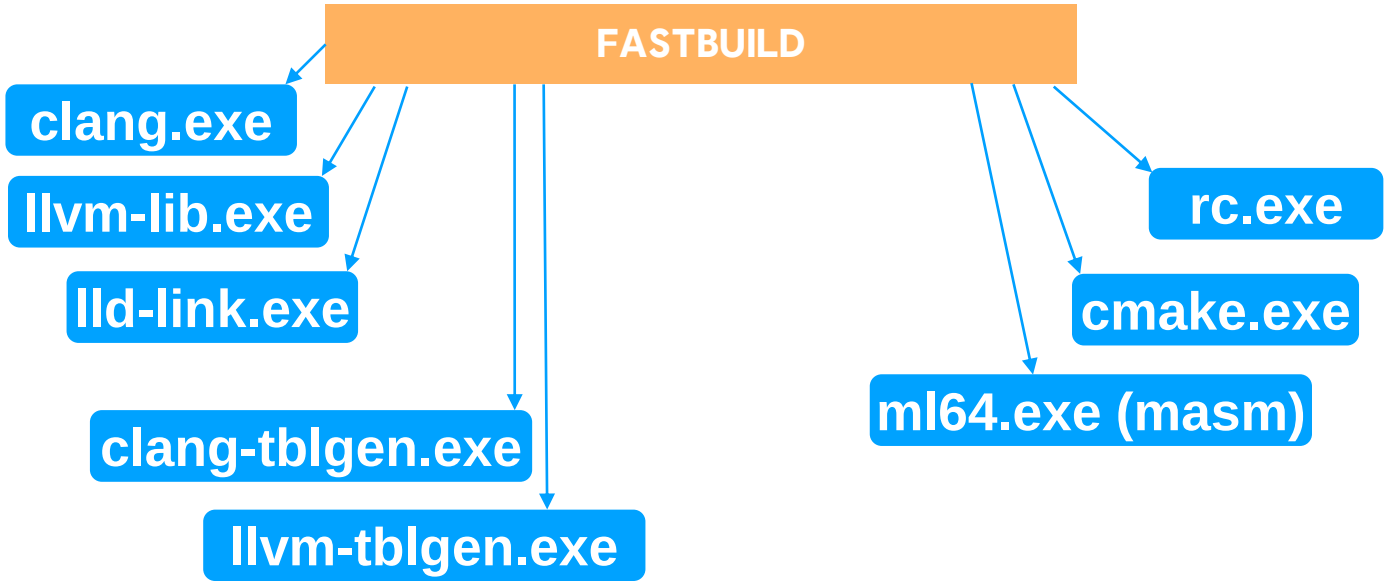


PART 3

PROPOSAL

PROOF-OF-CONCEPT

PREVIOUS BUILD PROCESS

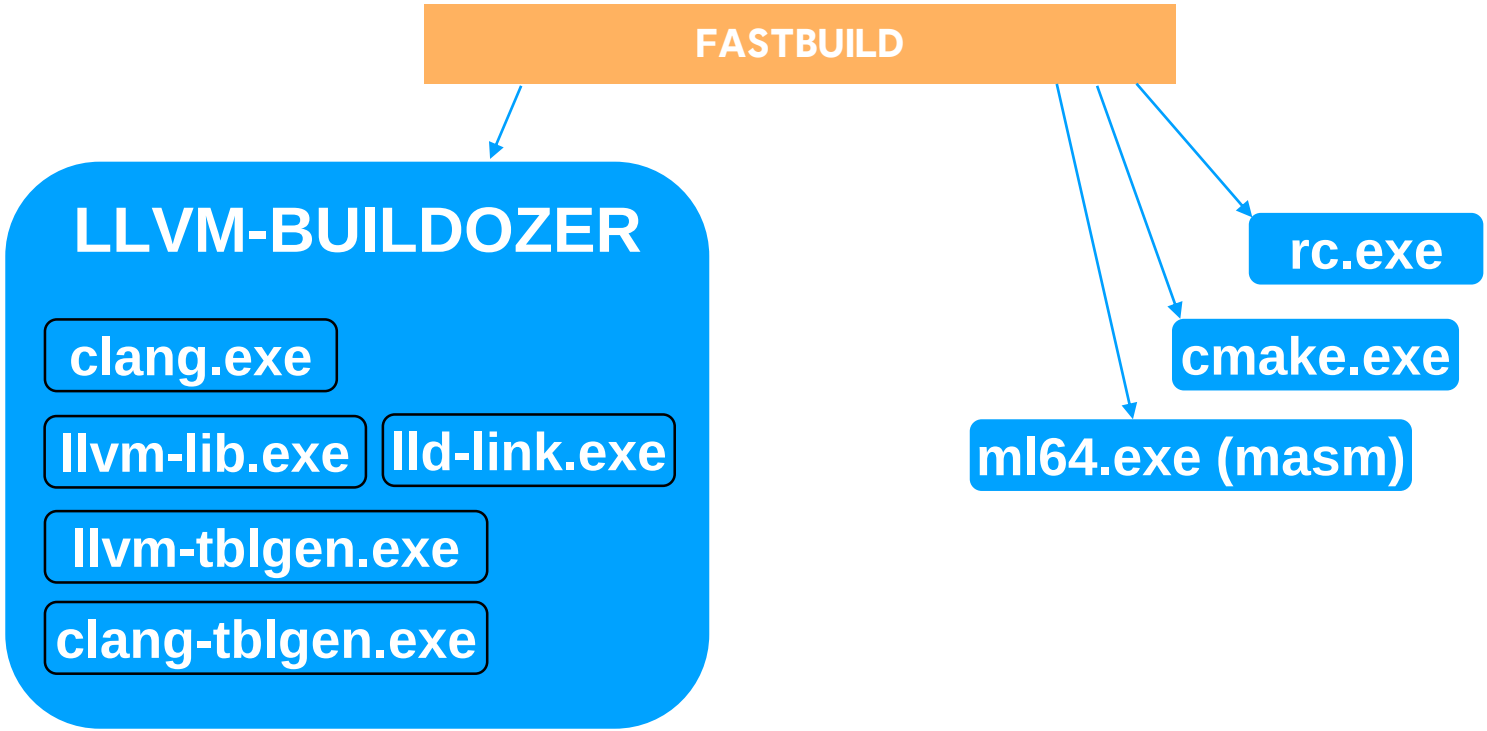


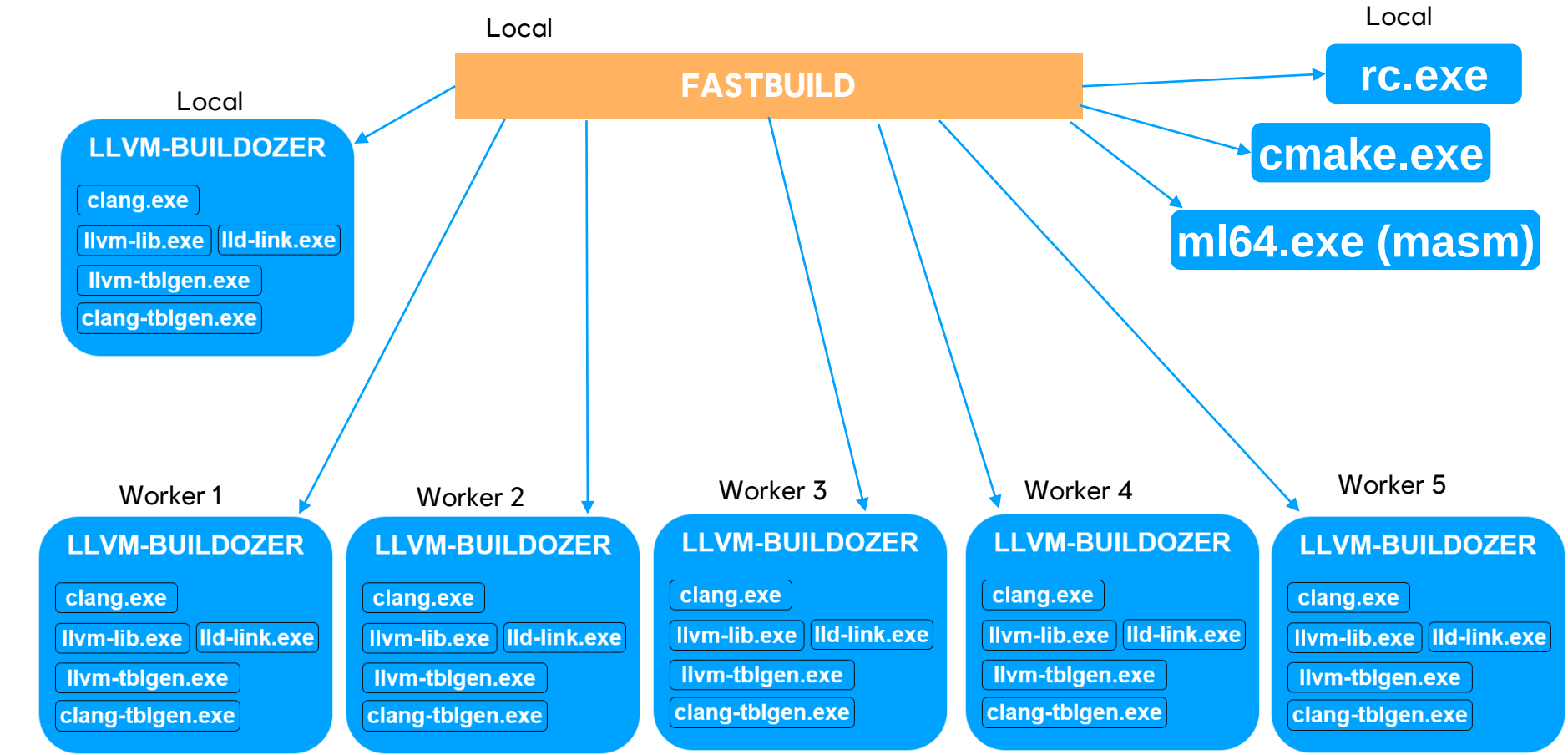
Maybe there’s a better way

LLVM-BUILDOZER

Image Credit: Caterpillar

BUILDING WITH BUILDOZER




Local
FASTBUILD
Local
rc.exe
cmake.exe
ml64.exe (masm)
Local
LLVM-BUILDOZER
clang.exe
llvm-lib.exe
lld-link.exe
llvm-tblgen.exe
clang-tblgen.exe
Worker 1
LLVM-BUILDOZER
clang.exe
llvm-lib.exe
lld-link.exe
llvm-tblgen.exe
clang-tblgen.exe
Worker 2
LLVM-BUILDOZER
clang.exe
llvm-lib.exe
lld-link.exe
llvm-tblgen.exe
clang-tblgen.exe
Worker 3
LLVM-BUILDOZER
clang.exe
llvm-lib.exe
lld-link.exe
llvm-tblgen.exe
clang-tblgen.exe
Worker 4
LLVM-BUILDOZER
clang.exe
llvm-lib.exe
lld-link.exe
llvm-tblgen.exe
clang-tblgen.exe
Worker 5
LLVM-BUILDOZER
clang.exe
llvm-lib.exe
lld-link.exe
llvm-tblgen.exe
clang-tblgen.exe

RUNNING THE DOZER

```
int buildozer::ImportEXE(llvm::StringRef EXE) {
  [..]
  HINSTANCE H = LoadLibraryA(EXE.data());
  if (!H)
    return 0;
```

"**LoadLibrary** can also be used to load other executable modules.[..]
However, do not use **LoadLibrary** to run an .exe file.
Instead, use the CreateProcess function." *(MSDN)*

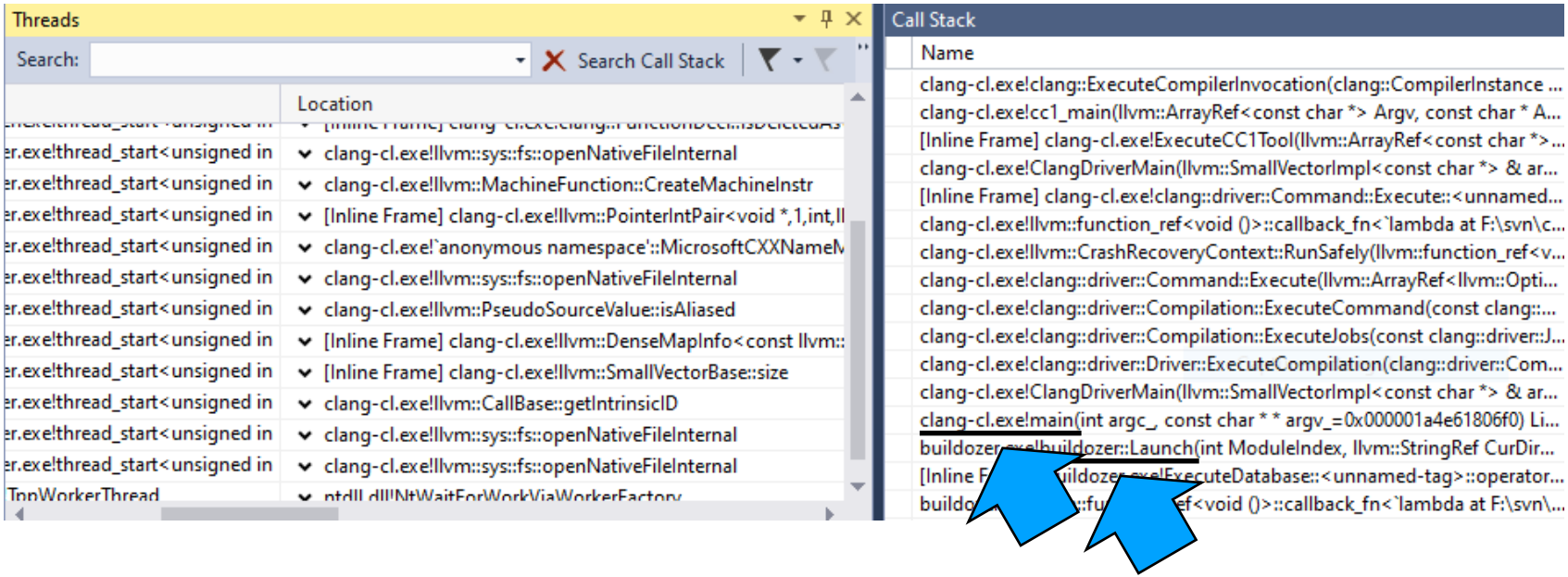
RUNNING THE DOZER

```
int buildozer::ImportEXE(llvm::StringRef EXE) {
  [..]
  HINSTANCE H = LoadLibraryA(EXE.data());
  if (!H)
    return 0;

  RemapImportAddressTable(H);
  InitDebInfo();
  PatchRPMalloc(M);
  InitializeStaticTLS(H);
  InitializeCRT(M);
  FindEntryPoints(M);
  [..]
}
```

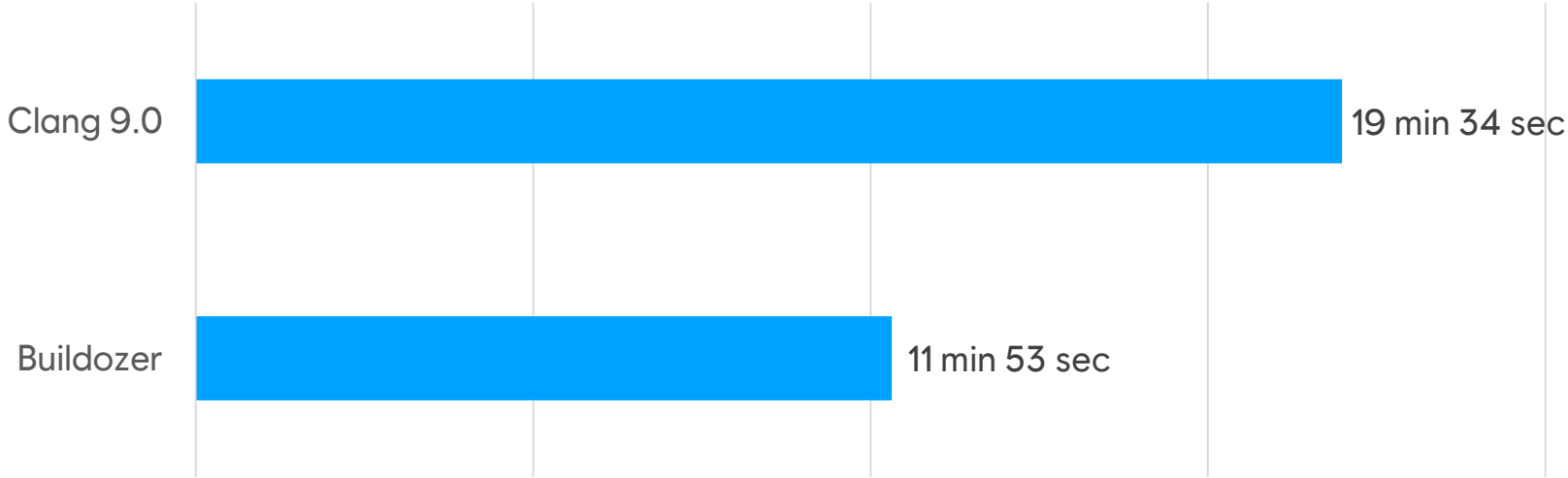
RUNNING THE DOZER

```
Pool.emplace(NumWorkers, [&]() {
  while (true) {
    buildozer::WorkUnit *WU = AcquireWork(..);
    if (!WU)
      break;

    int Mod = IdentifyMOD(WU);

    llvm::CrashRecoveryContext CRC;
    CRC.RunSafely([&] {
      buildozer::Launch(Mod, WU->Directory, WU->Arguments);
    });
    [..]
  }
});
Pool.join();
```

RUNNING THE DOZER



Local build, AAA game, x64 Editor Release



Intel Xeon W-2135 @ 3.7 GHz (6-core), 128 GB, NVMe SSD


PART 4
NEXT STEPS

SHORT TERM

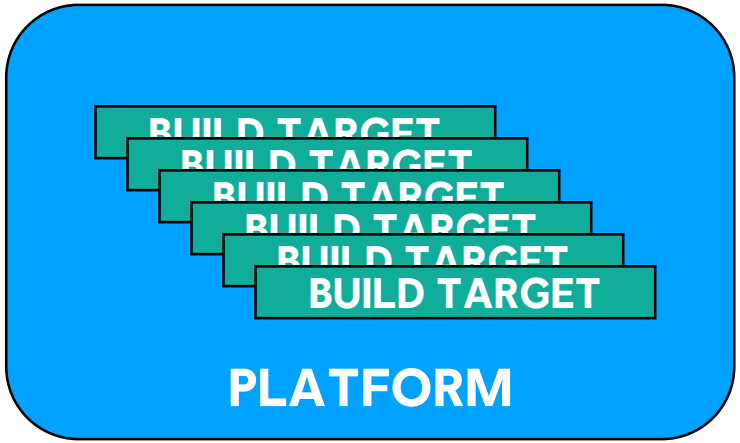
- Remove OS jitter (in-RAM file content & stat cache)
- OBJ cache (in-RAM)
- Clang-LLD in-memory bridge

- Incrementally link along the way
- Incrementally compile along the way (SN Systems' Program Repository)
- Remote API for distribution & caching

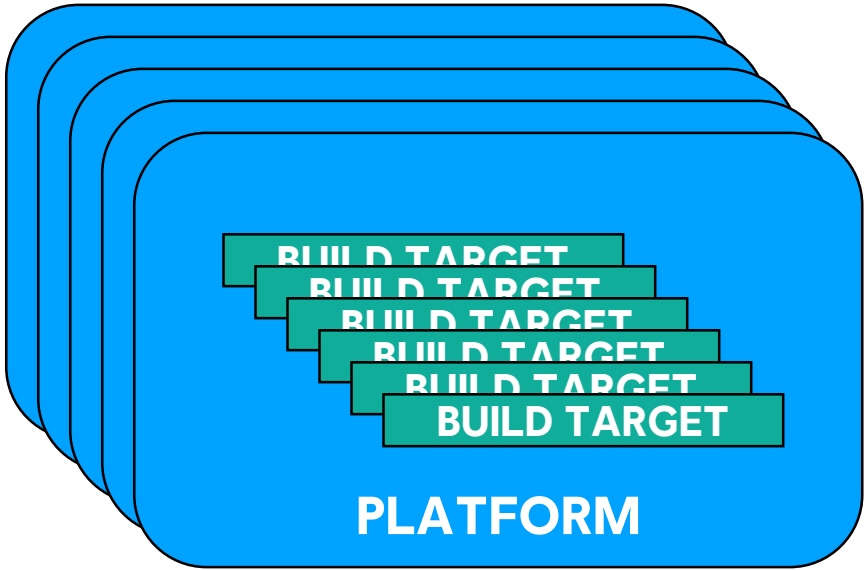
LONG TERM

BUILD TARGET

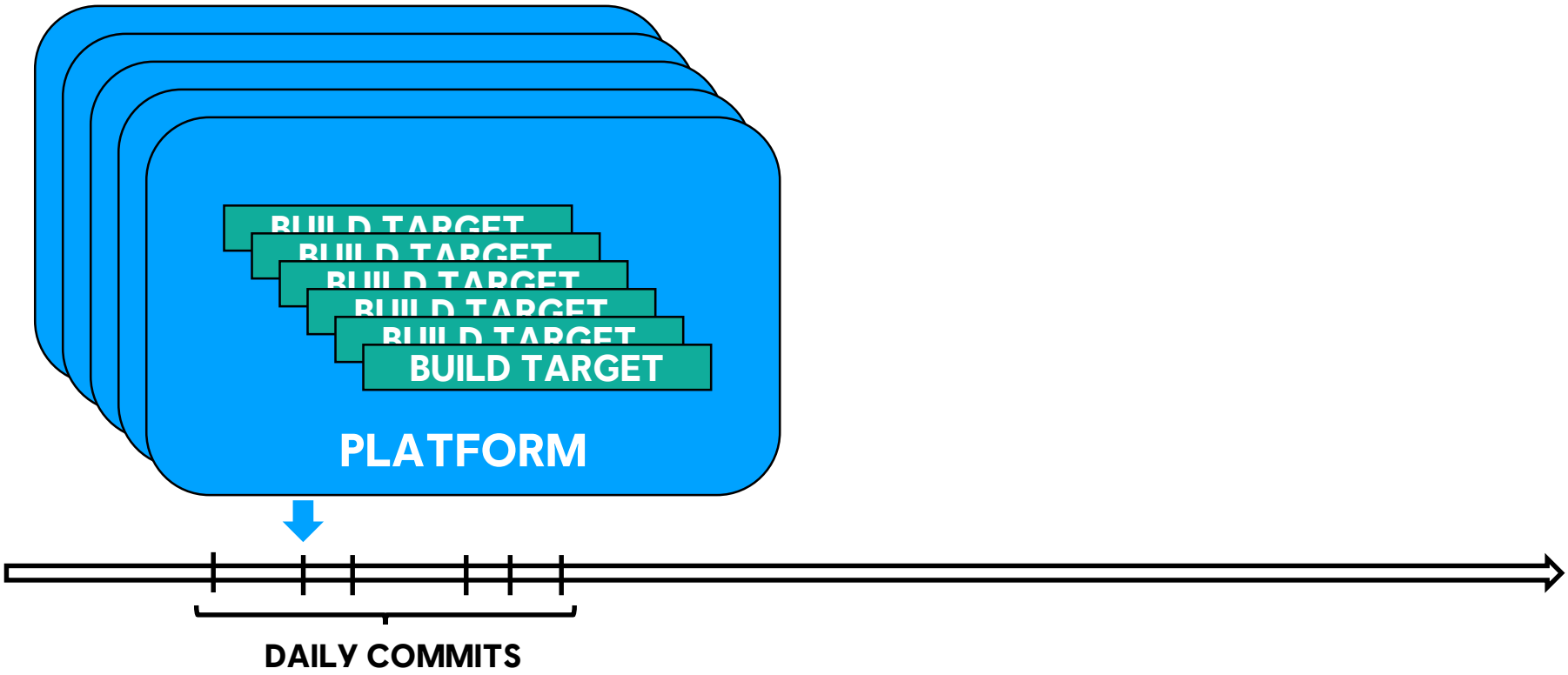
LONG TERM


BUILD TARGET
BUILD TARGET
BUILD TARGET
BUILD TARGET
BUILD TARGET
BUILD TARGET
PLATFORM

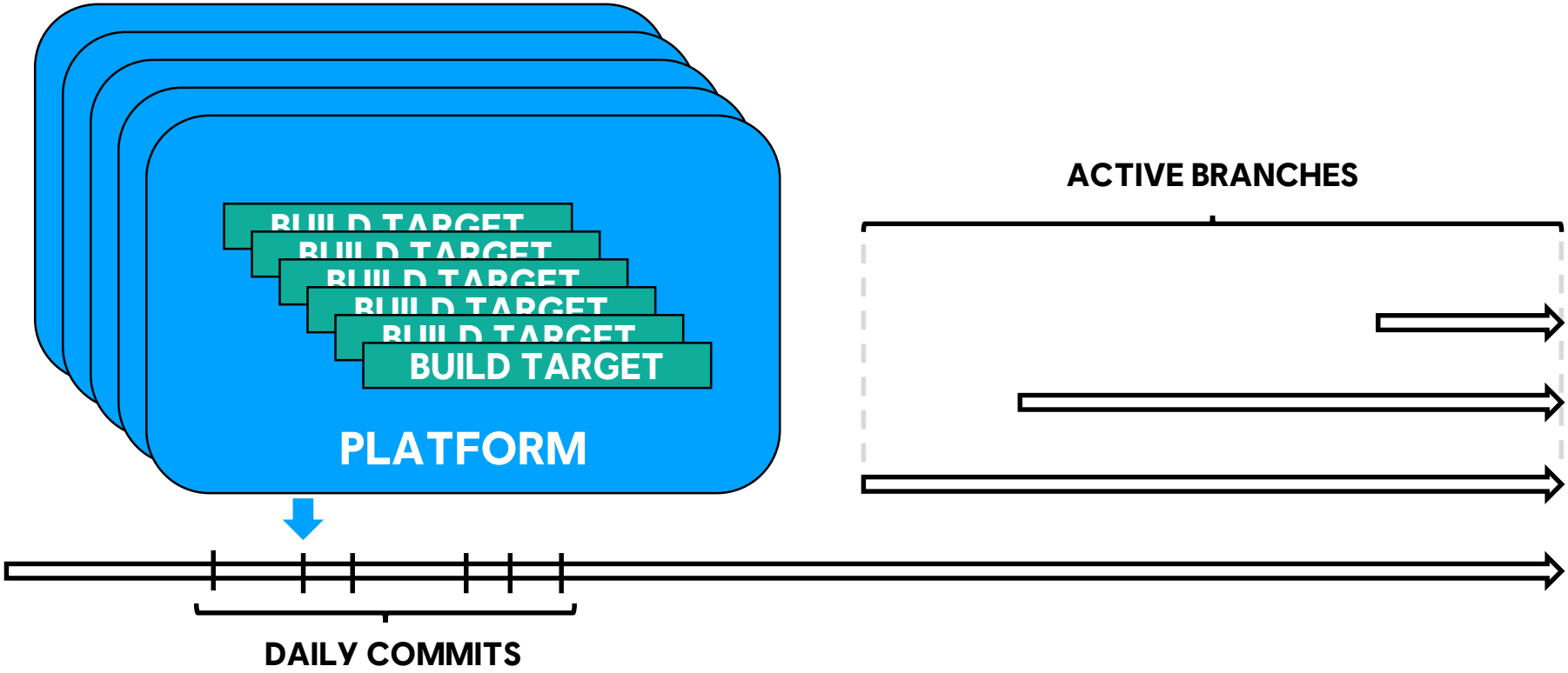
LONG TERM



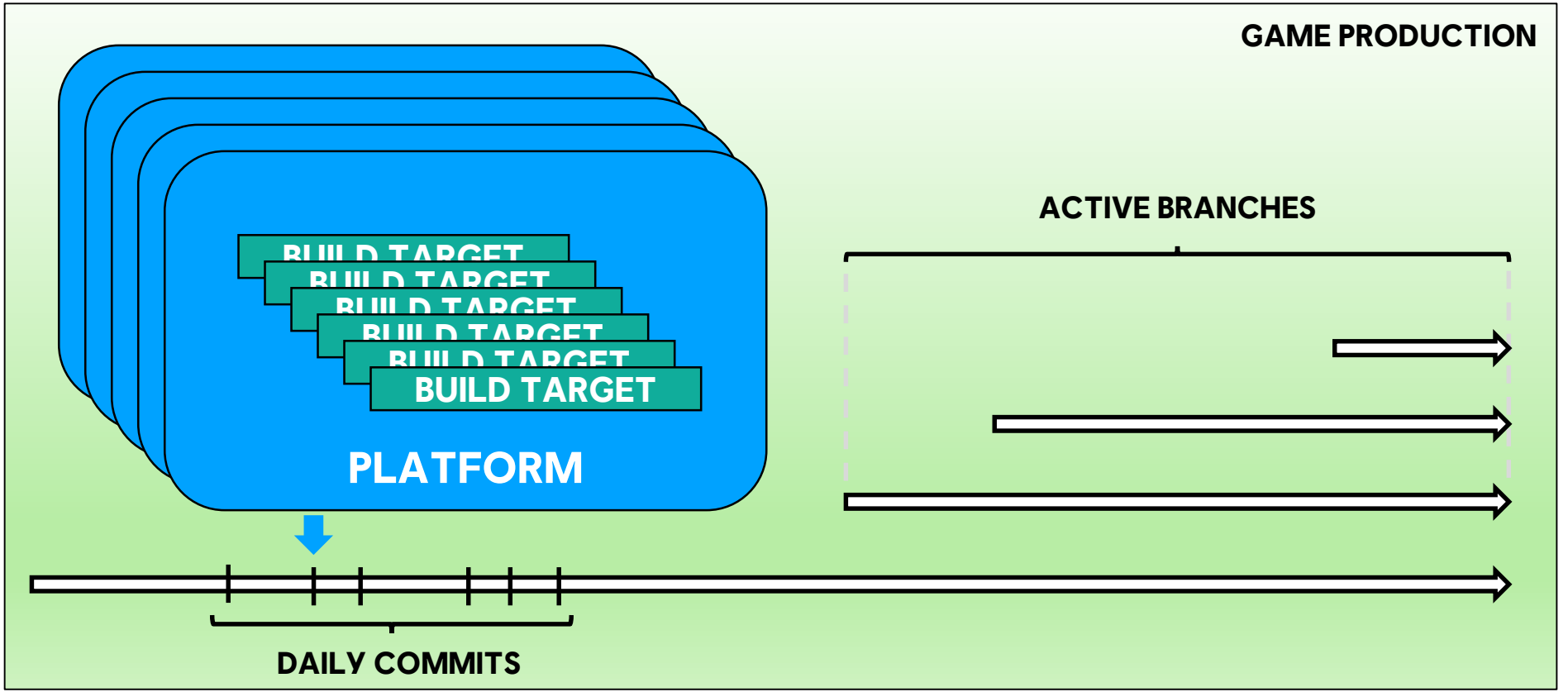
LONG TERM



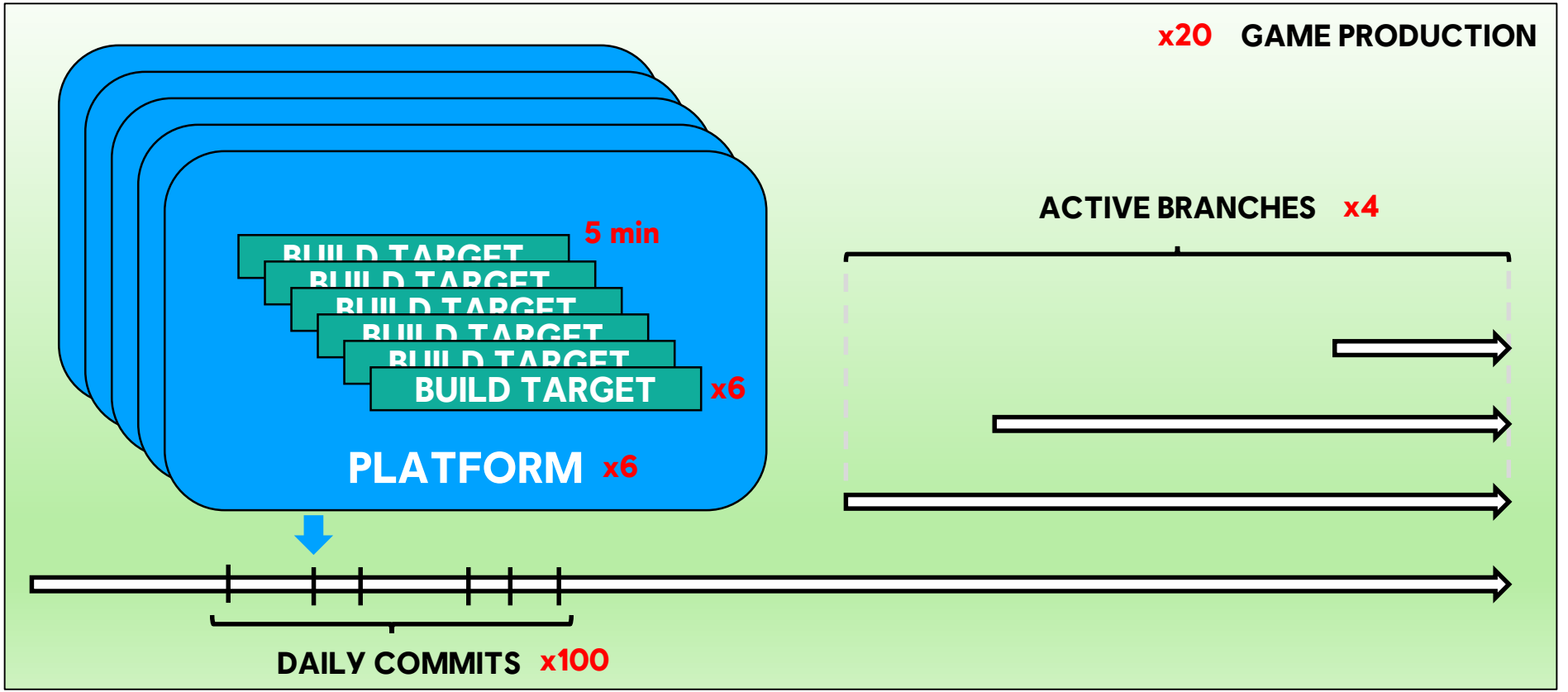
LONG TERM


BUILD TARGET
BUILD TARGET
BUILD TARGET
BUILD TARGET
BUILD TARGET
BUILD TARGET
PLATFORM
ACTIVE BRANCHES
DAILY COMMITS

LONG TERM


GAME PRODUCTION
BUILD TARGET
BUILD TARGET
BUILD TARGET
BUILD TARGET
BUILD TARGET
BUILD TARGET
PLATFORM
ACTIVE BRANCHES
DAILY COMMITS

LONG TERM


x20 GAME PRODUCTION
5 min
BUILD TARGET
BUILD TARGET
BUILD TARGET
BUILD TARGET
BUILD TARGET
BUILD TARGET
x6
PLATFORM x6
ACTIVE BRANCHES x4
DAILY COMMITS x100

Is there a better way?


THANK YOU

Q&A

Alexandre Ganea, Ubisoft
alexandre.ganea@ubisoft.com