Subject: Different native pthread.h implementations
Posted by Sender Ghost on Fri, 21 Jan 2011 06:04:25 GMT
View Forum Message <> Reply to Message

Hello.

There is following error on FreeBSD for multithreading source code, after 3041, 3044 revisions:

Web/socket.cpp: In function 'Upp::String& Upp::SockErrorText()':
Web/socket.cpp:815: error: cast from 'pthread*' to 'int' loses precision

Native implementations of pthread.h are different on BSD operating systems:
DragonFlyBSD:

```
/*
 * Run-time invariant values:
 */
#define PTHREAD_DESTRUCTOR_ITERATIONS  4
#define PTHREAD_KEYS_MAX   256
#define PTHREAD_STACK_MIN   1024
#define PTHREAD_THREADS_MAX   ULONG_MAX
#define PTHREAD_BARRIER_SERIAL_THREAD  -1
```

FreeBSD:

```
/*
 * Run-time invariant values:
 */
#define PTHREAD_DESTRUCTOR_ITERATIONS  4
#define PTHREAD_KEYS_MAX   256
#define PTHREAD_STACK_MIN   __MINSIGSTKSZ
#define PTHREAD_THREADS_MAX   __ULONG_MAX
#define PTHREAD_BARRIER_SERIAL_THREAD  -1
```

OpenBSD:

```
/*
 * Run-time invariant values:
 */
#define PTHREAD_DESTRUCTOR_ITERATIONS  4
#define PTHREAD_KEYS_MAX   256
#define PTHREAD_STACK_MIN   2048
#define PTHREAD_THREADS_MAX   ULONG_MAX
```

Not sure about NetBSD.

I solved it with following source code changes:

```
diff -ruN uppsrc/Web/socket.cpp uppsrc-fixed/Web/socket.cpp
--- uppsrc/Web/socket.cpp 2011-01-21 09:20:01.000000000 +0600
+++ uppsrc-fixed/Web/socket.cpp 2011-01-21 09:23:27.000000000 +0600
@@ -812,7 +812,11 @@
  int tid = GetCurrentThreadId();
 #else
  #ifdef _MULTITHREADED
-  int tid = (int)Thread::GetCurrentId();
+  #ifdef PLATFORM_BSD
+   unsigned long tid = (unsigned long)Thread::GetCurrentId();
+  #else
+   int tid = (int)Thread::GetCurrentId();
+  #endif
  #else
   int tid = 0;
  #endif
```

May be it possible to use "unsigned long" for all implementations.

Subject: Re: Different native pthread.h implementations
Posted by mirek on Fri, 21 Jan 2011 14:16:35 GMT
View Forum Message <> Reply to Message

Sender Ghost wrote on Fri, 21 January 2011 01:04
May be it possible to use "unsigned long" for all implementations.

Would 'uintptr_t' work for you?

Mirek

Subject: Re: Different native pthread.h implementations
Posted by mirek on Fri, 21 Jan 2011 14:17:18 GMT
View Forum Message <> Reply to Message

Other than that, that whole piece of code is not very nice, so perhaps the REAL solution is to use TLS there...

Subject: Re: Different native pthread.h implementations
Posted by Sender Ghost on Fri, 21 Jan 2011 14:52:31 GMT
View Forum Message <> Reply to Message

mirek wrote on Fri, 21 January 2011 15:16
Would 'uintptr_t' work for you?

Mirek

Yes, it compiles on FreeBSD.

uintptr_t tid = (uintptr_t)Thread::GetCurrentId();

Thanks.

Subject: Re: Different native pthread.h implementations
Posted by mirek on Fri, 21 Jan 2011 17:45:20 GMT
View Forum Message <> Reply to Message

Well, I have came to conlusion that all that solution of error management of Socket is desperately wanting, so I will completely refactor that function out tomorrow...

Subject: Re: Different native pthread.h implementations
Posted by tojocky on Sat, 22 Jan 2011 09:42:21 GMT
View Forum Message <> Reply to Message

Sender Ghost wrote on Fri, 21 January 2011 16:52mirek wrote on Fri, 21 January 2011 15:16
Would 'uintptr_t' work for you?

Mirek

Yes, it compiles on FreeBSD.

uintptr_t tid = (uintptr_t)Thread::GetCurrentId();

Thanks.

Maybe:

Thread::Id tid = Thread::GetCurrentId();
?

Subject: Re: Different native pthread.h implementations
Posted by Sender Ghost on Sat, 22 Jan 2011 10:24:24 GMT

tojocky wrote on Sat, 22 January 2011 10:42
Maybe:

Thread::Id tid = Thread::GetCurrentId();
?

Then there are problems for thread_index Index container:

Web/socket.cpp: In function 'Upp::String& Upp::SockErrorText()':
Web/socket.cpp:824: error: invalid conversion from 'pthread*' to 'int'
Web/socket.cpp:824: error: initializing argument 1 of 'int Upp::AIndex<T, V, HashFn>::Find(const T&)
const [with T = int, V = Upp::Vector<int>, HashFn = Upp::StdHash<int>]'
Web/socket.cpp:827: error: invalid conversion from 'pthread*' to 'int'
Web/socket.cpp:827: error: initializing argument 1 of 'void Upp::AIndex<T, V, HashFn>::Add(const T&)
[with T = int, V = Upp::Vector<int>, HashFn = Upp::StdHash<int>]'

You need to create GetHashValue function for this type, even if you declare Index like following:

static Index<Thread::Id> thread_index;

In other words, there is needed an value, not a pointer to it.

Subject: Re: Different native pthread.h implementations
Posted by mirek on Sat, 22 Jan 2011 12:24:10 GMT

OK, I guess we need temporary solution quite fast, so please check if this works for you:

```
static String& SockErrorText()
{
 static StaticCriticalSection csect;
 CriticalSection::Lock lock(csect);
 static Index<uintptr_t> thread_index;
 static Array<String> thread_errors;
 static Vector<int> error_ticks;
 int t = msecs();
 if(thread_index.GetCount() >= 1000) {
  for(int i = thread_index.GetCount(); --i >= 0;)
   if(t - error_ticks[i] >= 60000) {
    thread_index.Remove(i);
```

```
    thread_errors.Remove(i);
    error_ticks.Remove(i);
   }
  }
#ifdef PLATFORM_WIN32
 uintptr_t tid = GetCurrentThreadId();
#else
 #ifdef _MULTITHREADED
  uintptr_t tid = (uintptr_t)Thread::GetCurrentId();
 #else
  uintptr_t tid = 0;
 #endif
#endif
 int f = thread_index.Find(tid);
 if(f < 0) {
  f = thread_index.GetCount();
  thread_index.Add(tid);
  thread_errors.Add();
  error_ticks.Add();
 }
 error_ticks[f] = t;
 return thread_errors[f];
}
```

Mirek

Subject: Re: Different native pthread.h implementations
Posted by Sender Ghost on Sun, 23 Jan 2011 00:53:55 GMT
View Forum Message <> Reply to Message

mirek wrote on Sat, 22 January 2011 13:24OK, I guess we need temporary solution quite fast, so please check if this works for you

This was related to TheIDE, which tested with MT flag on FreeBSD operating system.
As you saw, I already found "temporary" solution. So, no need to hurry here. I appreciate your efforts, and sorry about some misunderstanding.
Paraphrasing, the main reason of this topic is report about cross-platform related issue with multithreaded source code (and some solution to solve it, at first).

Returning to topic, I can say that suggested source code compiled well on FreeBSD (and its partial version is tested second time here).

Subject: Re: Different native pthread.h implementations
Posted by tojocky on Sun, 23 Jan 2011 14:16:09 GMT
View Forum Message <> Reply to Message

Mirek,

The main reason of created Thread::GetCurrentId() is to add in Index<Thread::Id>.

I think that for FreeBSD need to change the method Thread::GetCurrentId().

What is your opinion?

Thank you in advance!
Ion.

mirek wrote on Sat, 22 January 2011 14:24OK, I guess we need temporary solution quite fast, so please check if this works for you:

```
static String& SockErrorText()
{
 static StaticCriticalSection csect;
 CriticalSection::Lock lock(csect);
 static Index<uintptr_t> thread_index;
 static Array<String> thread_errors;
 static Vector<int> error_ticks;
 int t = msecs();
 if(thread_index.GetCount() >= 1000) {
  for(int i = thread_index.GetCount(); --i >= 0;)
   if(t - error_ticks[i] >= 60000) {
    thread_index.Remove(i);
    thread_errors.Remove(i);
    error_ticks.Remove(i);
   }
 }
#ifdef PLATFORM_WIN32
 uintptr_t tid = GetCurrentThreadId();
#else
 #ifdef _MULTITHREADED
  uintptr_t tid = (uintptr_t)Thread::GetCurrentId();
 #else
  uintptr_t tid = 0;
 #endif
#endif
 int f = thread_index.Find(tid);
 if(f < 0) {
  f = thread_index.GetCount();
  thread_index.Add(tid);
  thread_errors.Add();
```

```
  error_ticks.Add();
 }
 error_ticks[f] = t;
 return thread_errors[f];
}
```

Mirek

Subject: Re: Different native pthread.h implementations
Posted by mirek on Tue, 25 Jan 2011 11:38:39 GMT
View Forum Message <> Reply to Message

I would not care for now. That whole SocketError function is terrible abnomination, it has to go.

Errors should be stored locally in Socket - that automatically solves all these issues.
