OTP team update

Code BEAM STO 16-17 May 2019

Kenneth Lundin Erlang/OTP team 2019-05-16

Outline

- Releases
- OTP 21 and 22 highlights
- OTP 23 and beyond

Release Principles

- 1 major release per year (21, 22, 23)
- 3 planned patch packages like 22.1, 22.2,..
- Unplanned patch packages in between
- maint branch = 22.x
- master branch = next major = 23.0

Releases

- **Current: OTP 22.0 (May 14)**
- Planned patch packages:
  - OTP 22.1 September
  - OTP 22.2 December
  - OTP 22.3 March 2020
- OTP 23.0 May 2020

Highlights in OTP 21.2-3

- New modules **`atomics`** and **`counters`**
- Efficient configuration data with **`persistent_term`**
- “Since” tags added in all documentation

Highlights in OTP 21.2-3
atomics and counters

Atomics: atomic operations towards mutable atomic variables.

Counters: built on atomics

No SW level locking, very efficient for concurrent access

significantly more efficient and scalable than

```
ets:update_counter(Tab, Key, UpdateOp) -> Result
```

```
1> Cref = counters:new(10, []).
{atomics,#Ref<0.3688.8573.87>}
2> counters:add(Cref, 1, 1).
ok
3> counters:get(Cref, 1).
1
```

Highlights in OTP 21.2-3
persistent_term

- **Think like this!** Write once, read many
- Instead of the generate module in runtime hack!
- Expensive put and low cost get
- No copying of the data on get

Example from the shell:

```
1> persistent_term:put(myapp_calls,
Cref).
ok
2> Cref =
persistent_term:get(myapp_calls).
{atomics,#Ref<0.3688.8573.87>}
3> counters:get(Cref, 1).
1
```

Highlights in OTP 21.2-3
"Since" tags added in all documentation

filter(Pred, MapOrIter) -> Map

OTP 18.0

Types

```
Pred = fun((Key, Value) -> boolean())
MapOrIter = #{Key => Value} | iterator(Key, Value)
Map = #{Key => Value}
```

Returns a map `Map` for which predicate `Pred` holds true in `MapOrIter`.

The call fails with a `{badmap,Map}` exception if `MapOrIter` is not a map or valid iterator, or with `badarg` if `Pred` is not a function of arity 2.

Example:

```
> M = #{a => 2, b => 3, c=> 4, "a" => 1, "b" => 2, "c" => 4},
  Pred = fun(K,V) -> is_atom(K) andalso (V rem 2) =:= 0 end,
  maps:filter(Pred,M).
#{a => 2,c => 4}
```

find(Key, Map) -> {ok, Value} | error

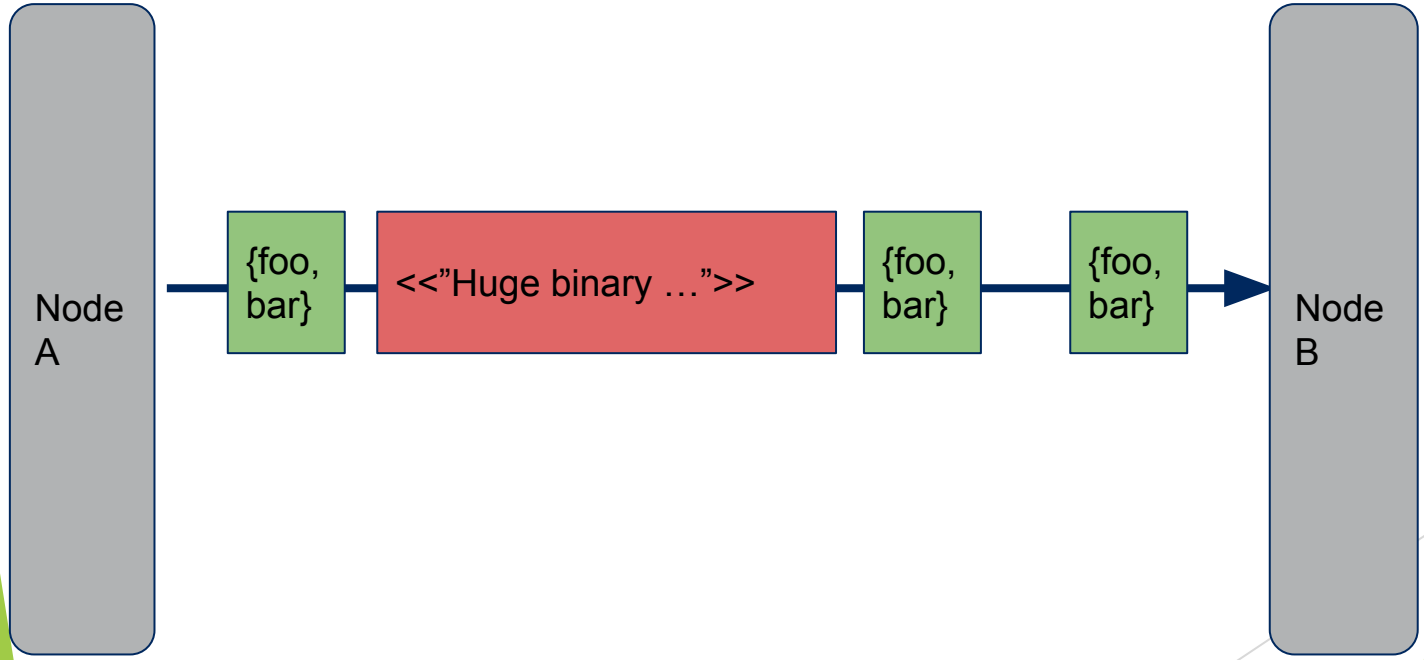
OTP 17.0

Highlights in OTP 22.0

- **ERTS**:
  - Improved memory handling
  - ETS `ordered_set` , `write_concurrency`
  - Socket NIF, experimental
  - Distribution protocol with fragmentation

Highlights in OTP 22.0
Distribution fragmentation

Solution for: Head of line blocking



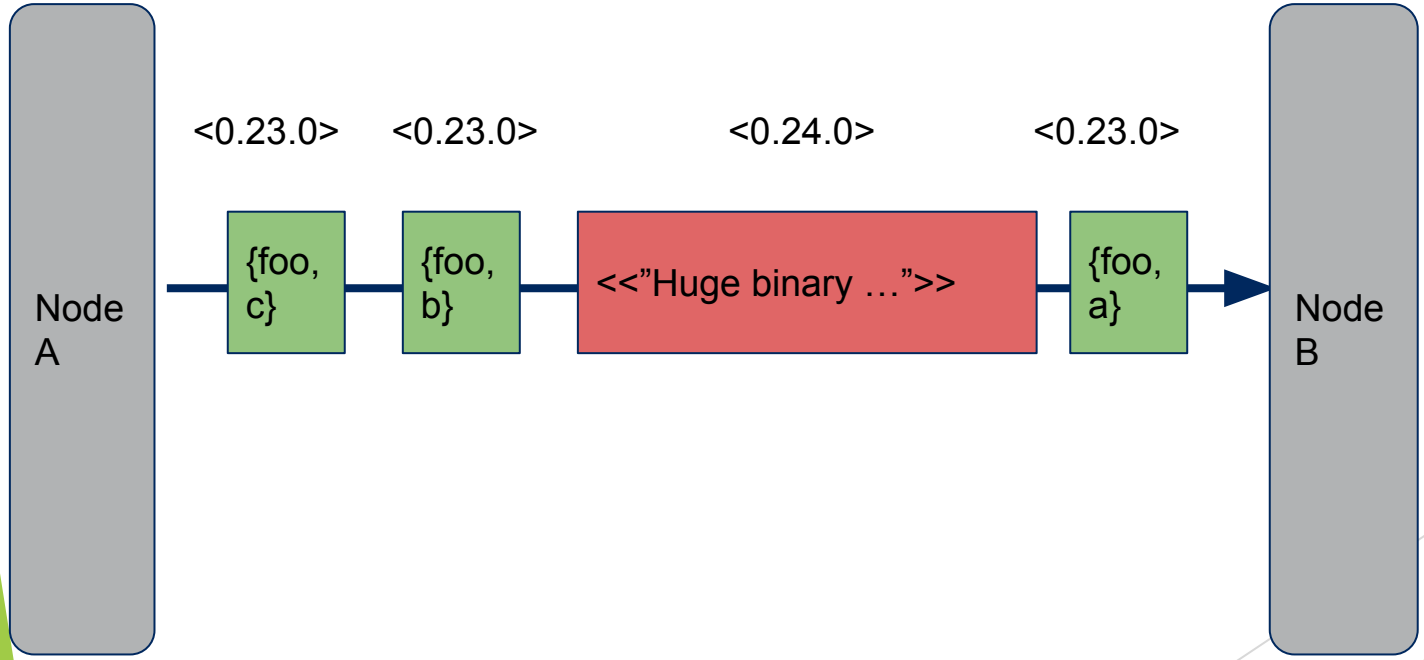
Highlights in OTP 22.0
Distribution fragmentation

Large Messages Between Nodes

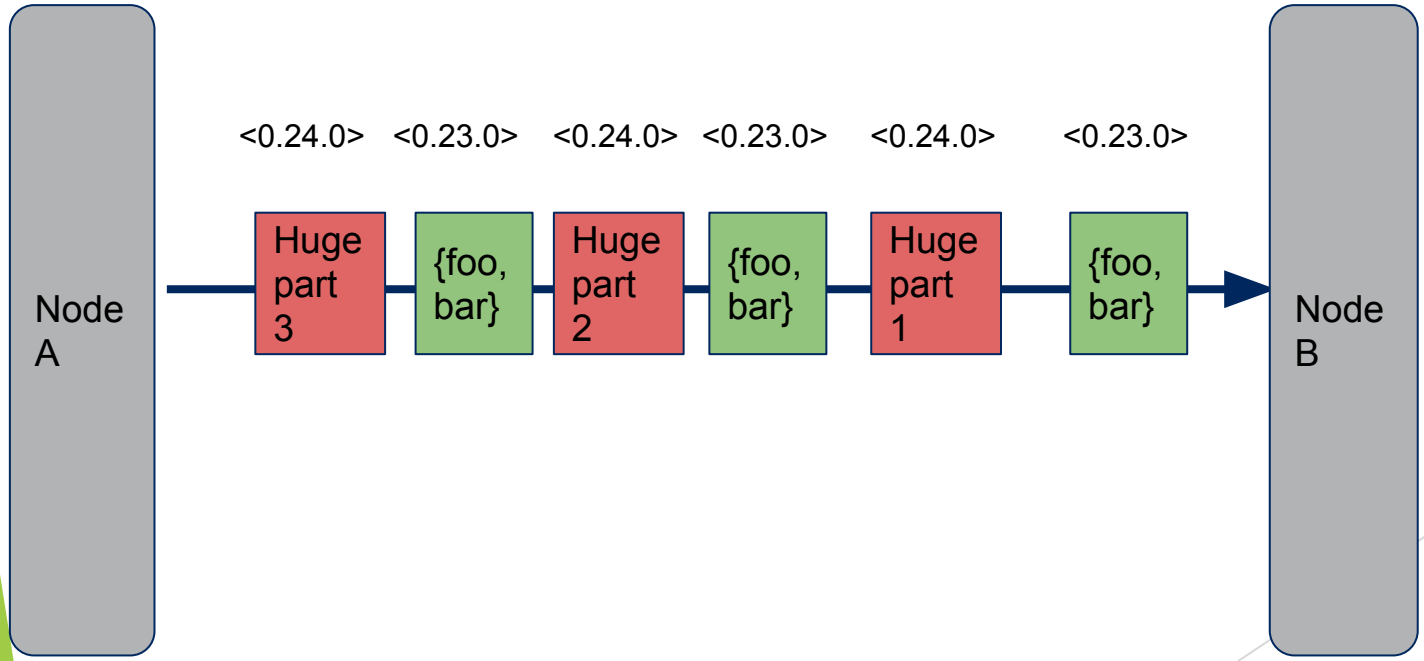
- !
- EXIT
- EXIT2
- MONITOR_DOWN

Highlights in OTP 22.0
Distribution fragmentation

Head of line blocking



Highlights in OTP 22.0
Distribution fragmentation



Highlights in OTP 22.0

- **Compiler**: optimizations using Static Single Assignment (SSA) representation
  - binary matching
  - better register usage
  - smaller stack frames
  - type analysis per module to remove unnecessary type checks in the runtime

Highlights in OTP 22.0
Compiler, optimizations, type analysis

```
f(A) when is_integer(A) ->
    {ok,A};
f(A) ->
    {error,{not_an_int,A}}.

g(A) ->
    case f(A) of
        {ok,Value} ->
            Value+1;
        {error,Reason} ->
            Reason
    end.
```

Highlights in OTP 22.0
Compiler, optimizations, type

```
f(A) when is_integer(A) ->
    {ok,A};
f(A) ->
    {error,{not_an_int,A}}.

g(A) ->
    case f(A) of
        {ok,Value} ->
            Value+1;
        {error,Reason} ->
            Reason
    end.
```

OTP 21

```
{function, g, 1, 5}.
...
  {allocate,0,1}.
  {call,1,{f,2}}.
  {test,is_tuple,{f,8},[{x,0}]}.
  {test,test_arity,{f,8},[{x,0},2]}.
  {get_tuple_element,{x,0},0,{x,1}}.
  {get_tuple_element,{x,0},1,{x,2}}.
  {test,is_atom,{f,8},[{x,1}]}.
  {select_val,{x,1},{f,8},
              {list,[{atom,error},{f,6},{atom,ok},{f,7}]}}.
 {label,6}.
  {move,{x,2},{x,0}}.
  {deallocate,0}.
  return.
 {label,7}.
  {gc_bif,'+',{f,0},3,[{x,2},{integer,1}],{x,0}}.
  {deallocate,0}.
  return.
 {label,8}.
  {case_end,{x,0}}.
```

Highlights in OTP 22.0
Compiler, optimizations, type

```
f(A) when is_integer(A) ->
    {ok,A};
f(A) ->
    {error,{not_an_int,A}}.

g(A) ->
    case f(A) of
        {ok,Value} ->
            Value+1;
        {error,Reason} ->
            Reason
    end.
```

OTP 22

```
{function, g, 1, 5}.
  …
    {allocate,0,1}.
    {call,1,{f,2}}.
    {get_tuple_element,{x,0},0,{x,1}}.
    {get_tuple_element,{x,0},1,{x,0}}.
    {test,is_eq_exact,{f,6},[{x,1},{atom,error}]}.
    {deallocate,0}.
    return.
{label,6}.
    {gc_bif,'+',{f,0},1,[{x,0},{integer,1}],{x,0}}.
    {deallocate,0}.
    return.
```

Highlights in OTP 22.0
Compiler, optimizations, type

OTP 21

```
{function, g, 1, 5}.
...
    {allocate,0,1}.
    {call,1,{f,2}}.
    {test,is_tuple,{f,8},[{x,0}]}.
    {test,test_arity,{f,8},[{x,0},2]}.
    {get_tuple_element,{x,0},0,{x,1}}.
    {get_tuple_element,{x,0},1,{x,2}}.
    {test,is_atom,{f,8},[{x,1}]}.
    {select_val,{x,1},{f,8},
                {list,[{atom,error},{f,6},{atom,ok},{f,7}]}}.
  {label,6}.
    {move,{x,2},{x,0}}.
    {deallocate,0}.
    return.
  {label,7}.
    {gc_bif,'+',{f,0},3,[{x,2},{integer,1}],{x,0}}.
    {deallocate,0}.
    return.
  {label,8}.
    {case_end,{x,0}}.
```

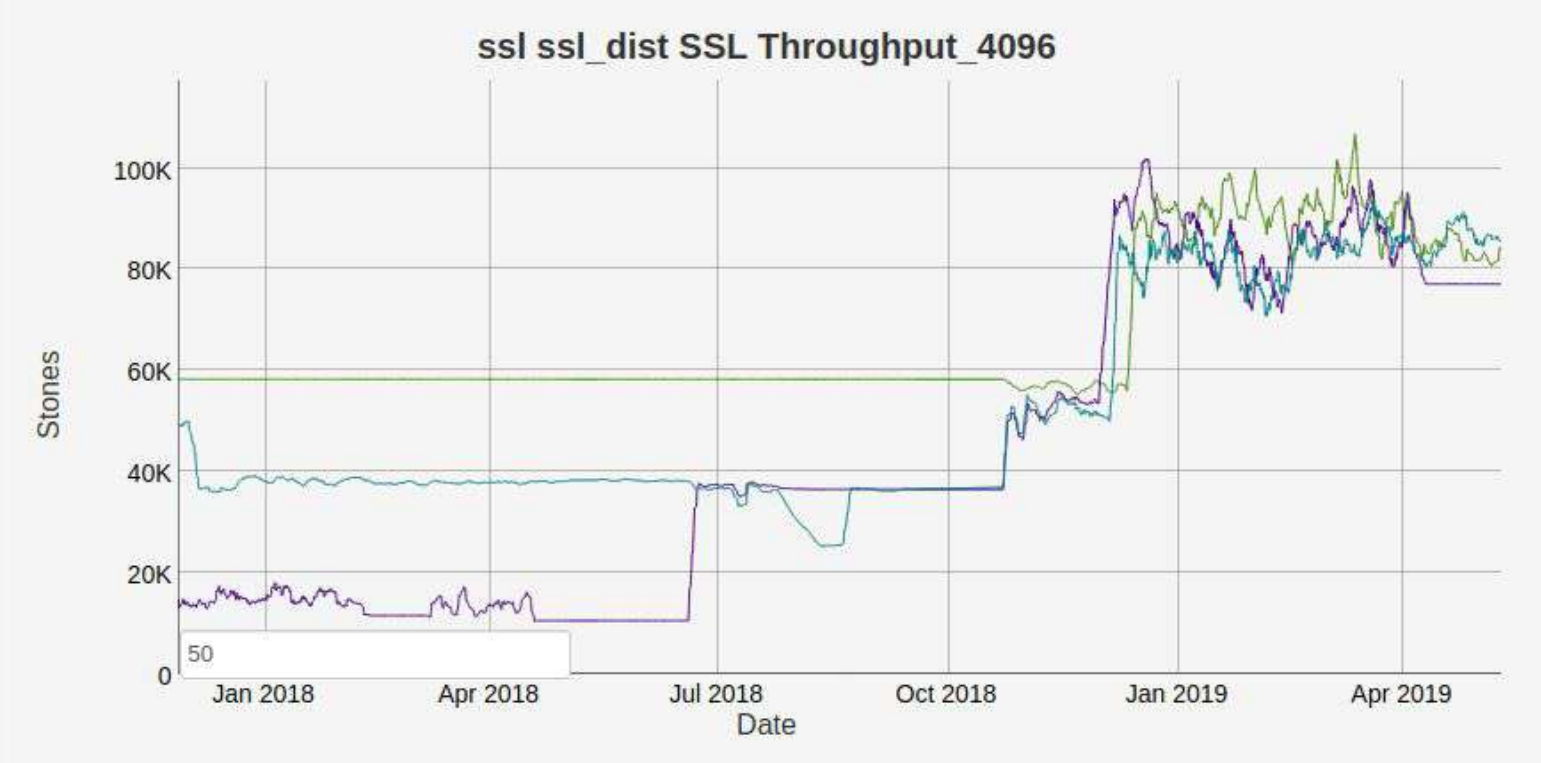
OTP 22

```
{function, g, 1, 5}.
  …
    {allocate,0,1}.
    {call,1,{f,2}}.
    {get_tuple_element,{x,0},0,{x,1}}.
    {get_tuple_element,{x,0},1,{x,0}}.
    {test,is_eq_exact,{f,6},[{x,1},{atom,error}]}.
    {deallocate,0}.
    return.
  {label,6}.
    {gc_bif,'+',{f,0},1,[{x,0},{integer,1}],{x,0}}.
    {deallocate,0}.
    return.
```

Highlights in OTP 22.0

- **SSL/crypto**:
  - TLS 1.3 server with limited functionality
  - TLS/DTLS logging ala OpenSSL (debug)
  - crypto rearranged structure
  - TLS performance improvements

Highlights in OTP 22.0
SSL/crypto



Highlights in OTP 22.0

- **Kernel:** Logger performance and features
- **Erl_interface:** plugin support, deprecate parts
- **Tools**: Cover ~2 times faster using `counters` and `persistent_term`

OTP 23 and beyond

- Compiler optimizations
- Distribution (network glitches, heterogenous, scalable, plug-able)
- Cloud, Container and micro service friendly
- socket NIFs + gen_tcp/udp/sctp based on that
- Continue with TLS 1.3 and TLS improvements in general
- Active in EEF work groups such as
  - Building and packaging, Observability, Security,
  - upcoming: Interoperability, Documentation
- JIT, ongoing, Open Source during this year.

Thanks for your contributions

- 127 different contributors the last year.
- Read more about the new features in OTP 22 in our blog here: http://blog.erlang.org/OTP-22-Highlights/

Questions?