Doing Blockchain with Elixir
The Good - The Bad - The Ugly

Dominic Letz / Diode CTO
Elixir Berlin meetup, September 2019

About Me

- Dominic Letz / 陳多米
- Co-Inventor of BlockQuick algorithm
- CTO of Diode, Exosite's new project
  https://diode.io
- Native Berliner but spent last 7 years in Taiwan

Founding Member of Ethereum Magicians Ring:
Constrained Resource Clients

PHP => C++ => Erlang => Elixir

Web3

Blockchain Based Decentralized REWRITE Of "The Internet"*

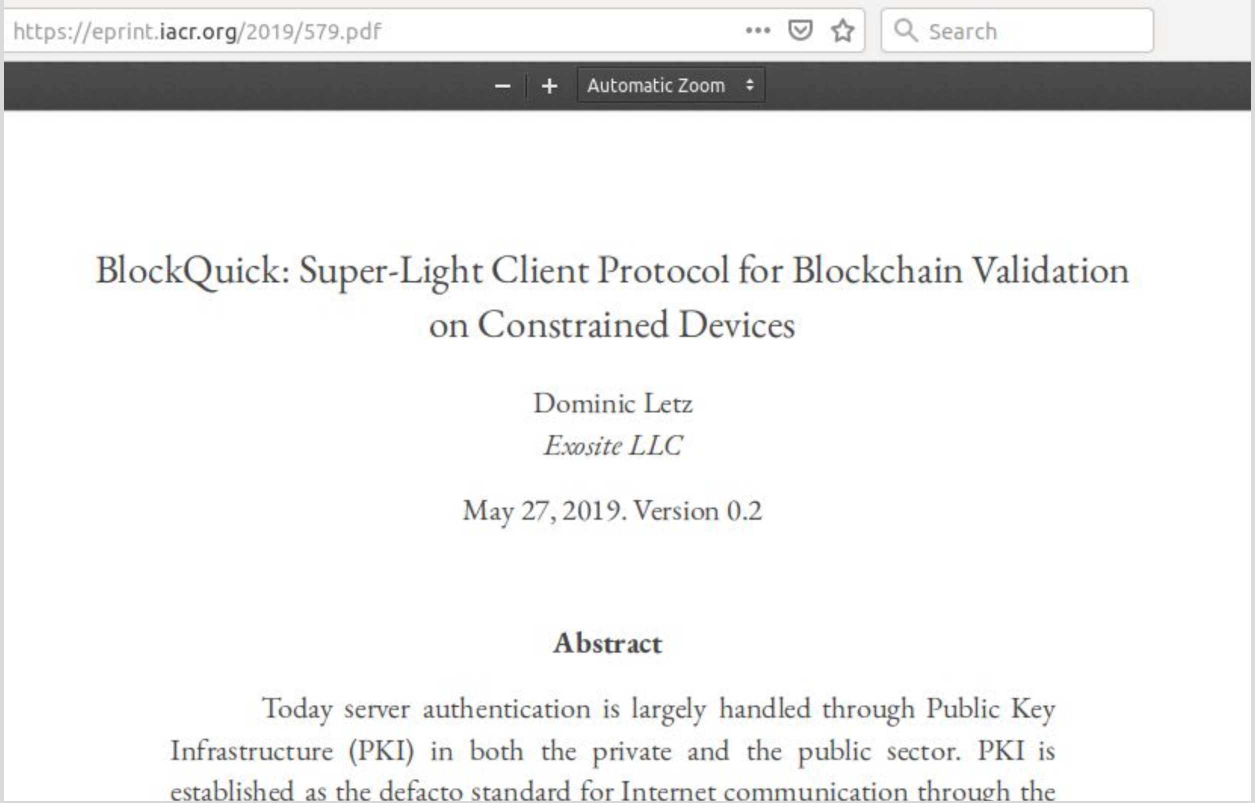
@LateNightSeth
THAT'S A FREAKIN' GREAT IDEA!!

Why Has Nobody Else Done That Yet?

Blockchain is Secure
But too Big for Clients

Client	Storage	RAM	Sync Bandwidth
geth --syncmode=fastsync	200 GB	1,000 MB	~100 MB per day
geth --syncmode=light	1.2 GB	150 MB	~3.5 MB per day
IOTA Node	8 GB	4,000 MB	1 GB per day

Hardware	Storage	RAM	Bandwidth
ESP32	4-16 MB	520 KB	WIFI
Linkit 7697	4 MB	352 KB	WIFI



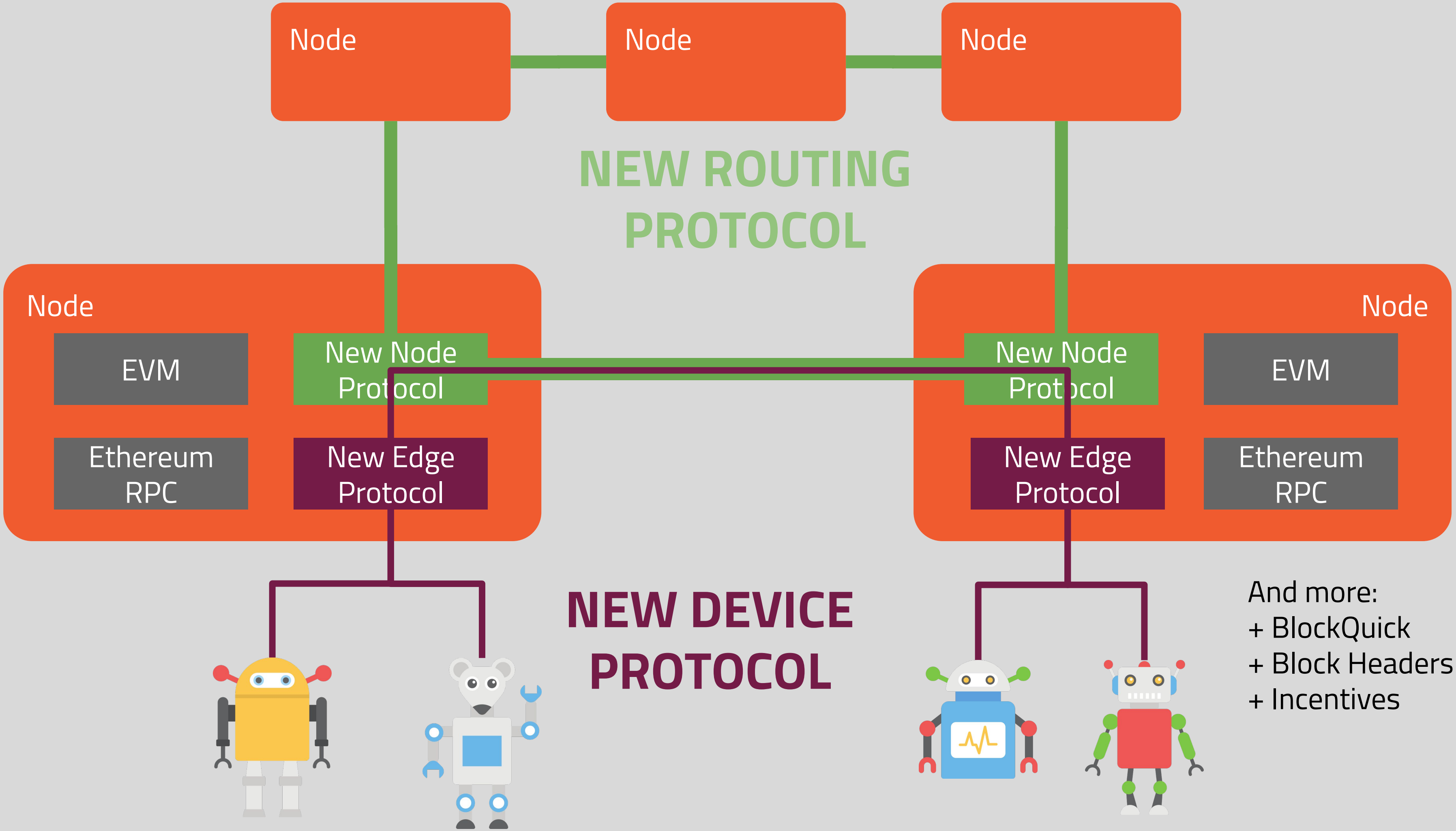
BlockQuick: Super-Light Client Protocol for Blockchain Validation on Constrained Devices

Dominic Letz
Exosite LLC

May 27, 2019. Version 0.2

Abstract

Today server authentication is largely handled through Public Key Infrastructure (PKI) in both the private and the public sector. PKI is established as the defacto standard for Internet communication through the



Node

Node

Node

NEW ROUTING PROTOCOL

Node

EVM

New Node Protocol

Ethereum RPC

New Edge Protocol

Node

New Node Protocol

EVM

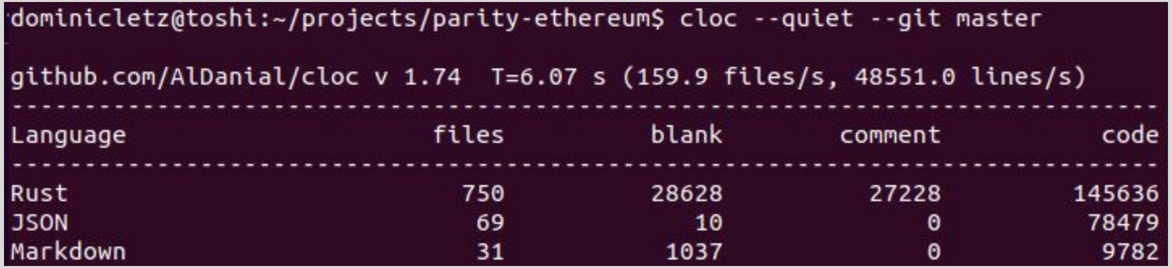
New Edge Protocol

Ethereum RPC

NEW DEVICE PROTOCOL

And more:
+ BlockQuick
+ Block Headers
+ Incentives

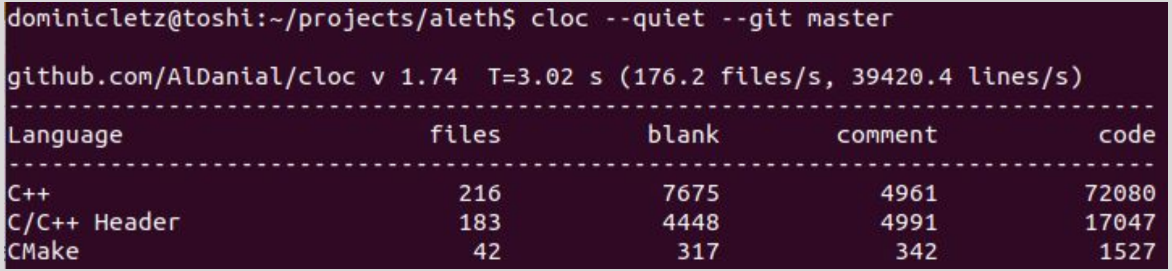
How much code do I need to read to understand Ethereum?



```
dominicletz@toshi:~/projects/parity-ethereum$ cloc --quiet --git master

github.com/AlDanial/cloc v 1.74  T=6.07 s (159.9 files/s, 48551.0 lines/s)
-------------------------------------------------------------------------------
Language                     files          blank        comment           code
-------------------------------------------------------------------------------
Rust                           750          28628          27228         145636
JSON                            69             10              0          78479
Markdown                        31           1037              0           9782
```

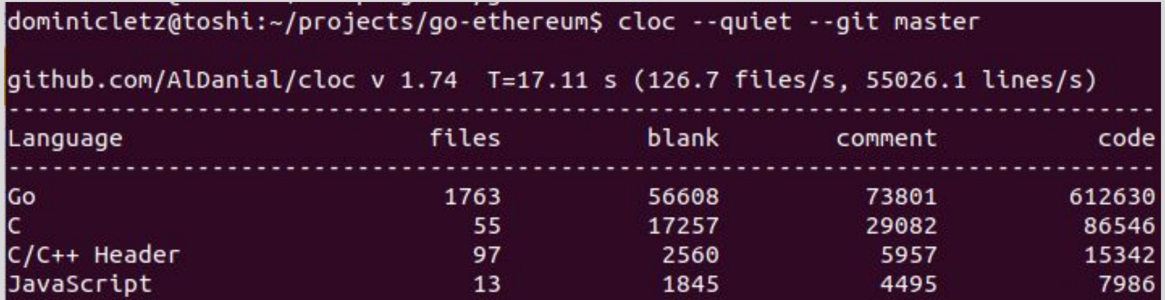
~145k Rust



```
dominicletz@toshi:~/projects/aleth$ cloc --quiet --git master

github.com/AlDanial/cloc v 1.74  T=3.02 s (176.2 files/s, 39420.4 lines/s)
-------------------------------------------------------------------------------
Language                     files          blank        comment           code
-------------------------------------------------------------------------------
C++                            216           7675           4961          72080
C/C++ Header                   183           4448           4991          17047
CMake                           42            317            342           1527
```

~89k C++



```
dominicletz@toshi:~/projects/go-ethereum$ cloc --quiet --git master

github.com/AlDanial/cloc v 1.74  T=17.11 s (126.7 files/s, 55026.1 lines/s)
-------------------------------------------------------------------------------
Language                     files          blank        comment           code
-------------------------------------------------------------------------------
Go                            1763          56608          73801         612630
C                               55          17257          29082          86546
C/C++ Header                    97           2560           5957          15342
JavaScript                      13           1845           4495           7986
```

~612k Go

Good: Many Places to Lend Pieces From

Erlang EVM: Aeternity
https://github.com/aeternity/aeternity

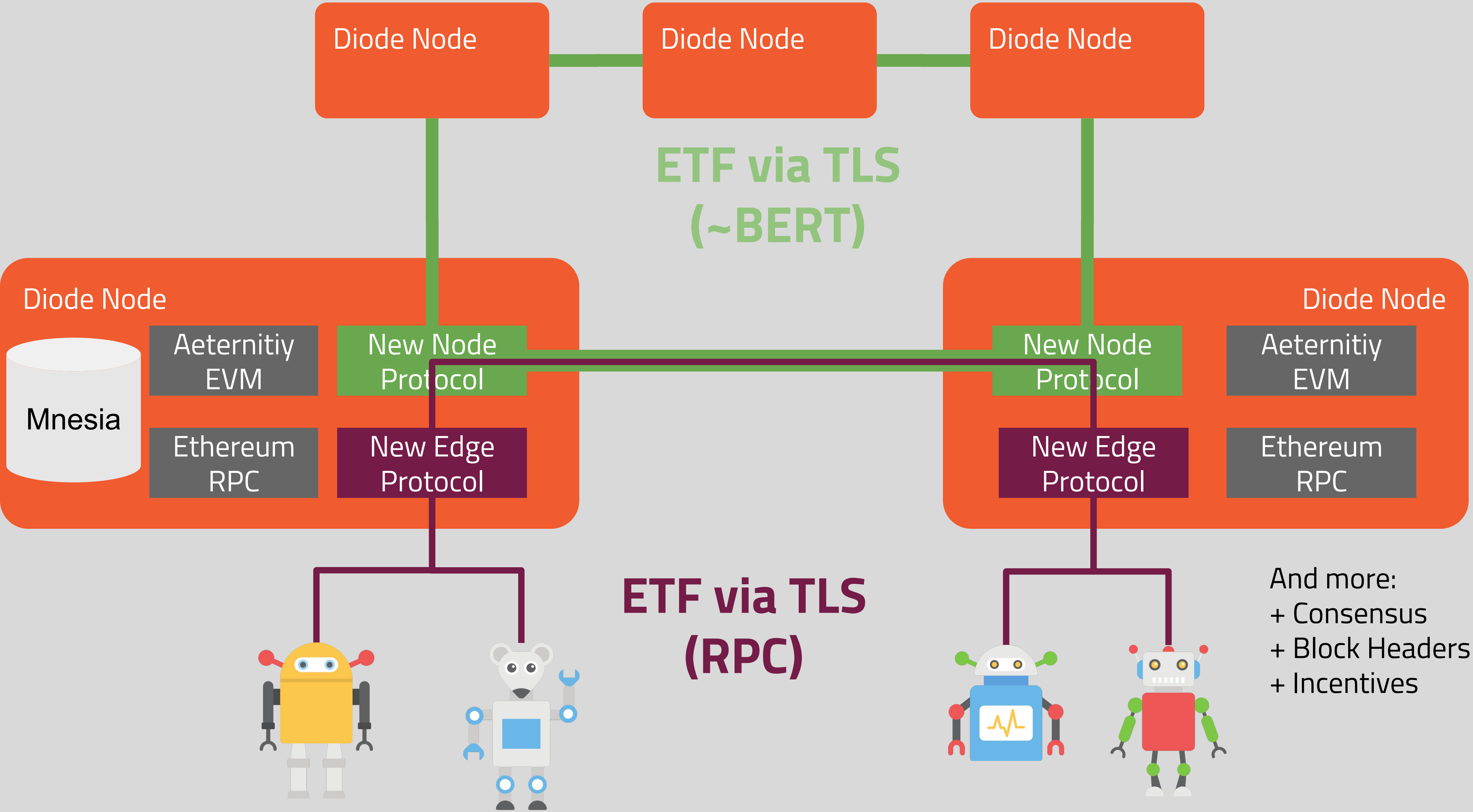
Elixir Network Explorer:
https://github.com/poanetwork/blockscout

Erlang secp256k1

https://hex.pm/packages/libsecp256k1

Elixir Full Node: Mana-Ethereum (not used)
https://github.com/mana-ethereum/mana

Elixir Prototype



Diode Node

Diode Node

Diode Node

ETF via TLS
(~BERT)

Diode Node

Mnesia

Aeternitiy
EVM

New Node
Protocol

Ethereum
RPC

New Edge
Protocol

Diode Node

New Node
Protocol

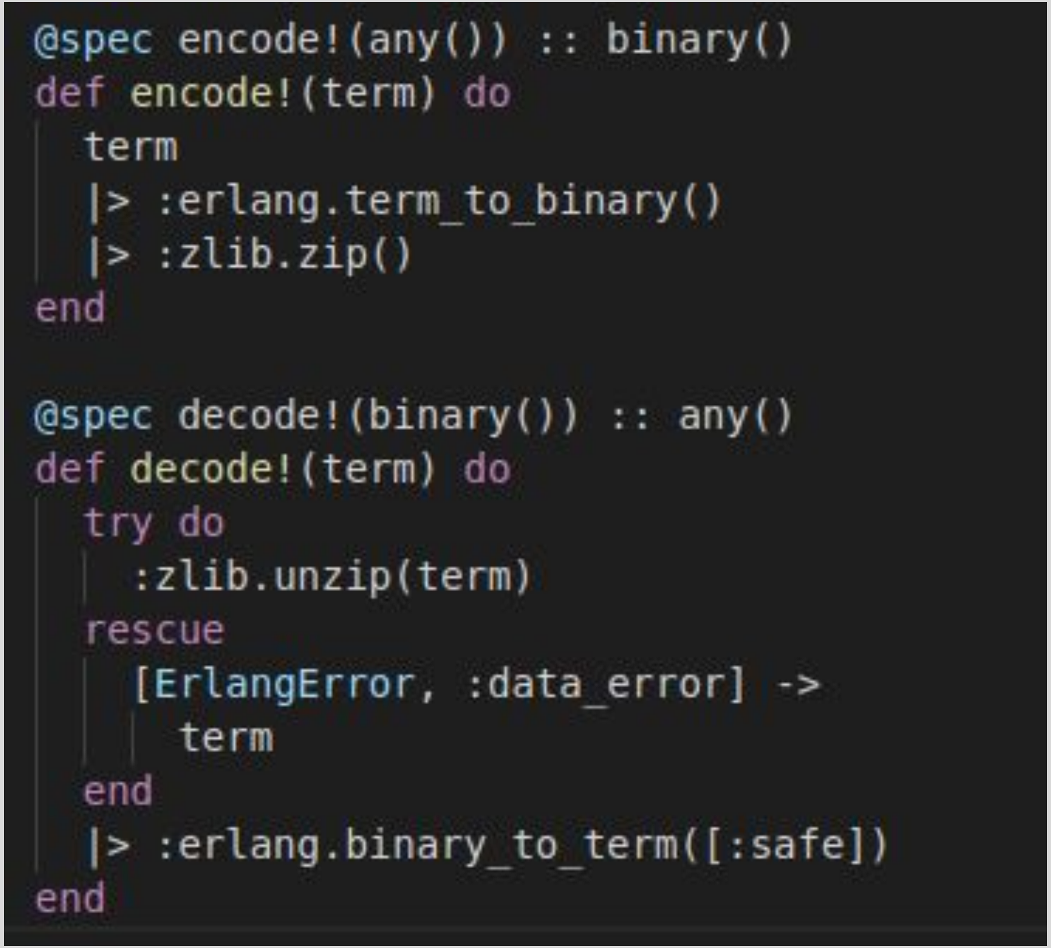
Aeternitiy
EVM

New Edge
Protocol

Ethereum
RPC

ETF via TLS
(RPC)

And more:
+ Consensus
+ Block Headers
+ Incentives

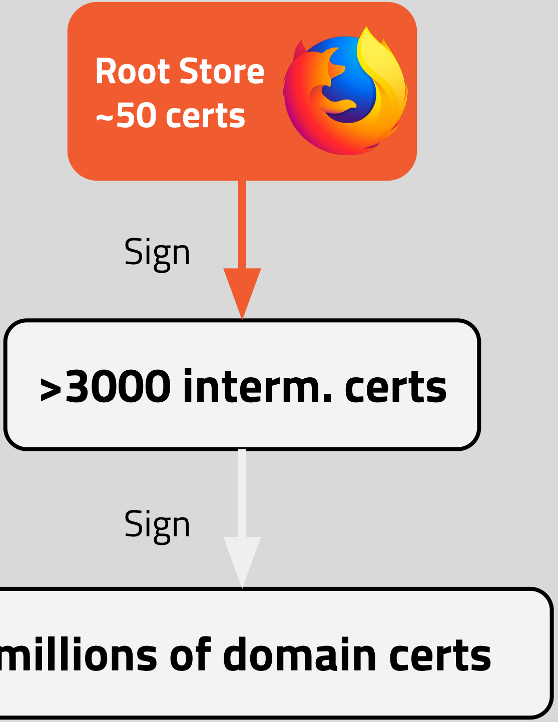


```elixir
@spec encode!(any()) :: binary()
def encode!(term) do
  term
  |> :erlang.term_to_binary()
  |> :zlib.zip()
end

@spec decode!(binary()) :: any()
def decode!(term) do
  try do
    :zlib.unzip(term)
  rescue
    [ErlangError, :data_error] ->
      term
  end
  |> :erlang.binary_to_term([:safe])
end
```

Today

Trust By Trusted Roots



Root Store
~50 certs

Sign

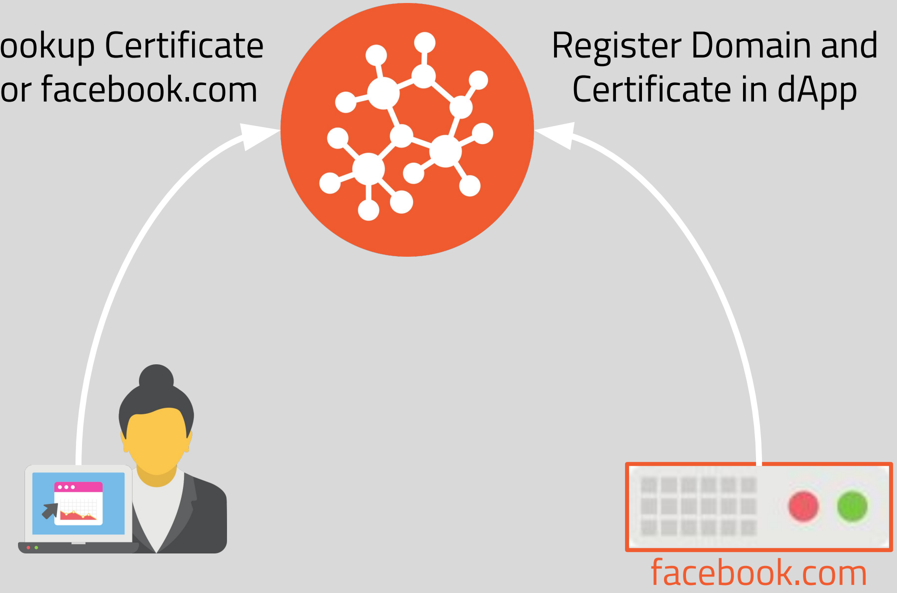
>3000 interm. certs

Sign

millions of domain certs

Blockchain

Trust By Consensus



Lookup Certificate
for facebook.com

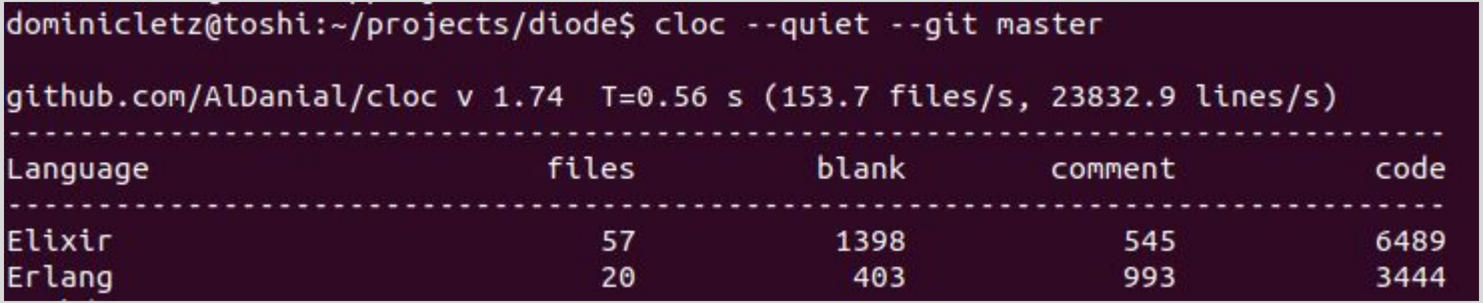
Register Domain and
Certificate in dApp

facebook.com

MITM impossible

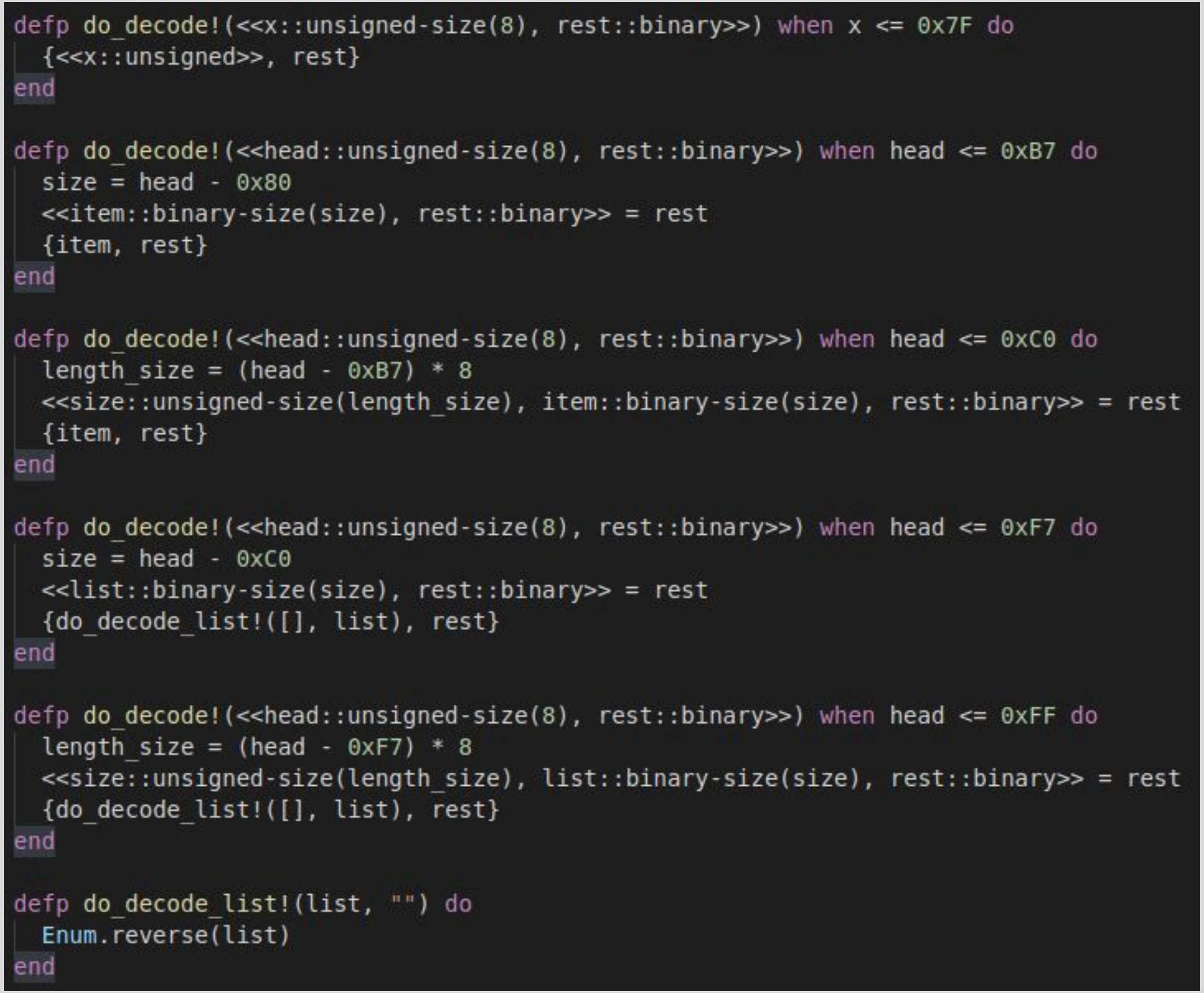
Writing your own Ethereum Node in Elixir

- Merkle Trees
- Recursive Data Structures like RLP
- Network Protocols
- Mnesia + ETS



```
dominicletz@toshi:~/projects/diode$ cloc --quiet --git master

github.com/AlDanial/cloc v 1.74  T=0.56 s (153.7 files/s, 23832.9 lines/s)
-------------------------------------------------------------------------------
Language                     files          blank        comment           code
-------------------------------------------------------------------------------
Elixir                          57           1398            545           6489
Erlang                          20            403            993           3444
```



```
defp do_decode!(<<x::unsigned-size(8), rest::binary>>) when x <= 0x7F do
  {<<x::unsigned>>, rest}
end

defp do_decode!(<<head::unsigned-size(8), rest::binary>>) when head <= 0xB7 do
  size = head - 0x80
  <<item::binary-size(size), rest::binary>> = rest
  {item, rest}
end

defp do_decode!(<<head::unsigned-size(8), rest::binary>>) when head <= 0xC0 do
  length_size = (head - 0xB7) * 8
  <<size::unsigned-size(length_size), item::binary-size(size), rest::binary>> = rest
  {item, rest}
end

defp do_decode!(<<head::unsigned-size(8), rest::binary>>) when head <= 0xF7 do
  size = head - 0xC0
  <<list::binary-size(size), rest::binary>> = rest
  {do_decode_list!([], list), rest}
end

defp do_decode!(<<head::unsigned-size(8), rest::binary>>) when head <= 0xFF do
  length_size = (head - 0xF7) * 8
  <<size::unsigned-size(length_size), list::binary-size(size), rest::binary>> = rest
  {do_decode_list!([], list), rest}
end

defp do_decode_list!(list, "") do
  Enum.reverse(list)
end
```

The Bad

You can't be 100% Elixir. Crypto routines will stay in C.

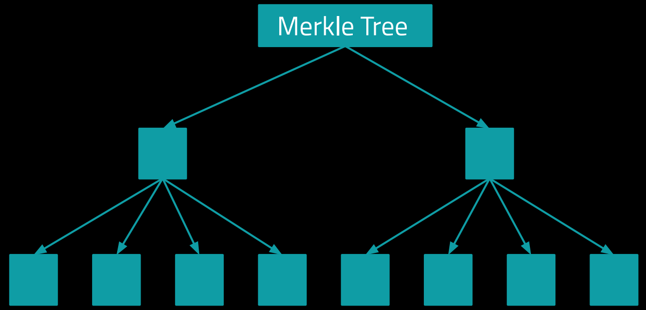
Don't Rewrite in Elixir!

If you do. Don't expect it to be nice or fast

https://github.com/dominicletz/exsha3/

The Ugly

Elixir is great to write SHORT CODE for (merkle) trees



But shared nothing means you have many copies, or only one process to work in the tree.

Diode

IOT SECURITY IS BROKEN MAKE IT ROCK SOLID

https://diode.io

https://github.com/diodechain

Get Involved