Building a universal secure computation platform based on protection domains

Dan Bogdanov, Roman Jagomägis,
Peeter Laud, Jaak Randmets, Jaak Ristioja

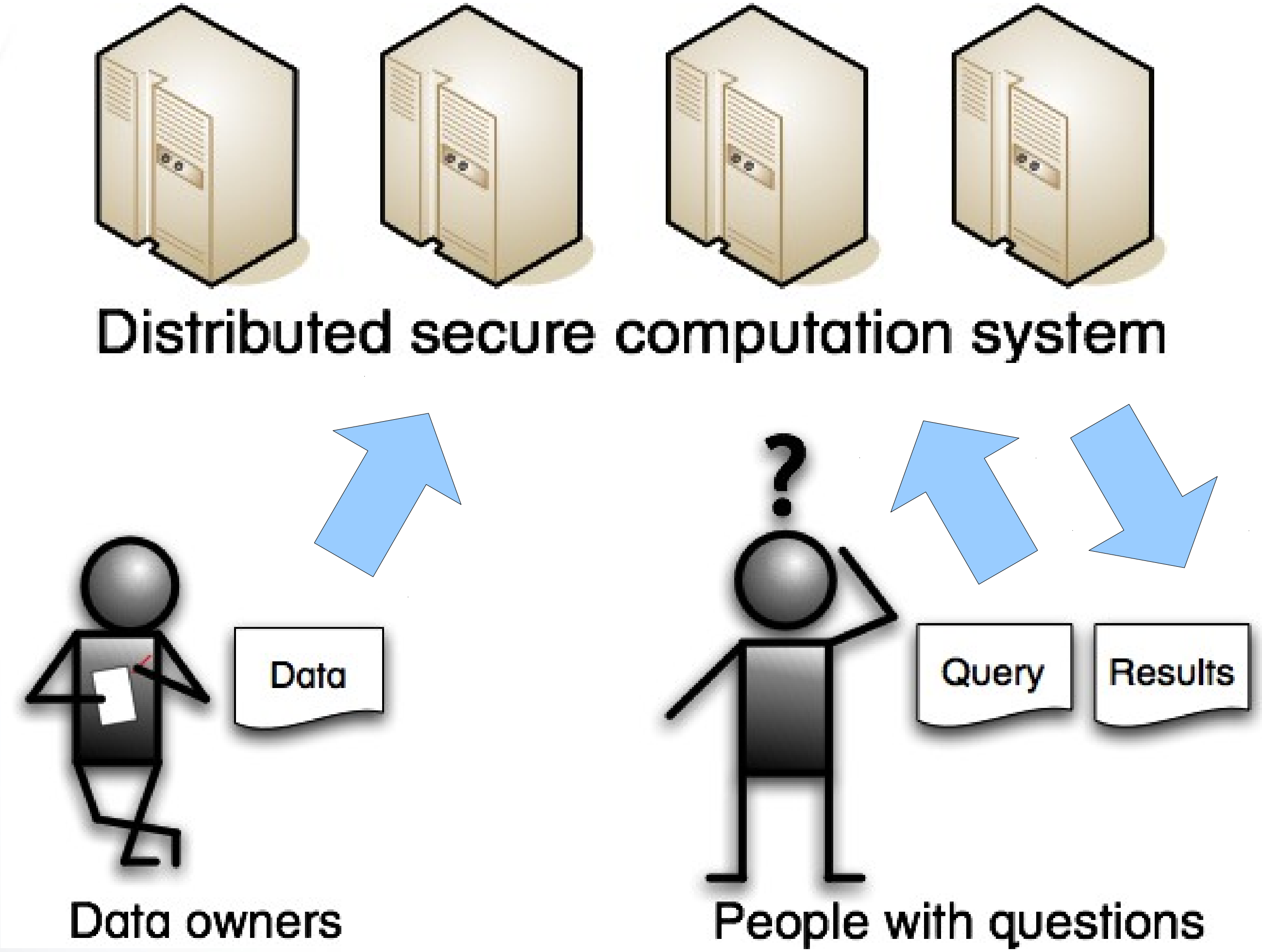
Cybernetica

Estonian Computer Science Theory Days in Kubija
January 27th -29th, 2012

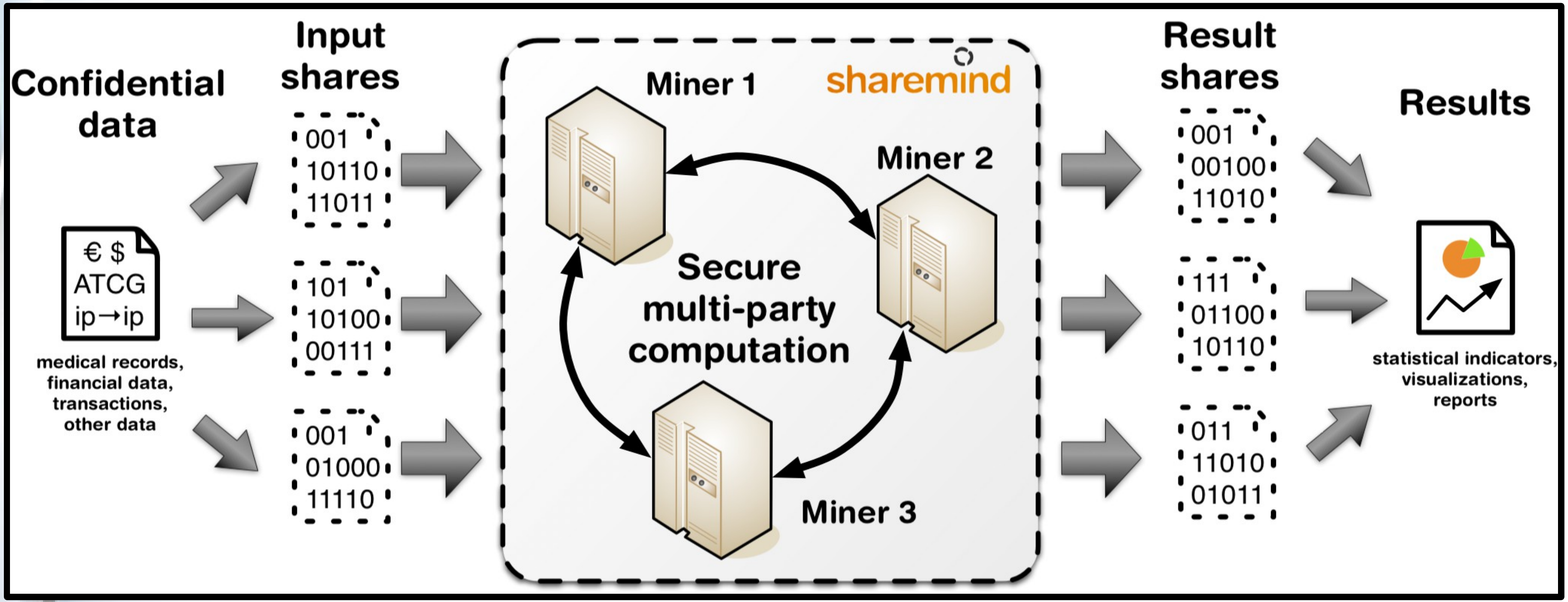
This research was, in part, funded by the U.S. Government. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Government.
“A” (Approved for Public Release, Distribution Unlimited)

CYBERNETICA

Basic scenario



Earlier work: Sharemind 2



How Sharemind works

- Algorithms for processing data are deployed on the servers.
- Client applications send in the data and query parameters.
- The server performs secure computation on the provided data.
- The server returns the results to the client application who made the query.

Earlier work: SecreC

```
// SecreC is an algorithm language with
// built-in visibility supertypes private and public
// private values become public through declassify

void main () {                 // main function
  private int a, b, c;         // private data
  a = b + c;                   // private computation
  public int d;                // public data
  d = declassify (a);          // make private public
  publish (d);                 // send to client
}
```

Sharemind 3 & SecreC 2

- We need to be able to quickly adapt new secure computation techniques and security models.
- These techniques should be easily accessible in the SecreC language.

- We are updating both the virtual machine and the language.

Sharemind 3 design goals

- The virtual computer acts like an *ideal functionality* or a *trusted party*.
- The computer can support various secure computing technologies.
- The computer has features for both short-term and long-term storage.
- The computer can also work with public data.

Protection domains

A *protection domain* (PD) is a set of data that is protected with the same resources and for which there is a well-defined set of algorithms and protocols for computing on that data while keeping the protection.

Protection domain kinds

A *protection domain kind* is a set of data representations, algorithms and protocols for storing and computing on protected data.

Each protection domain belongs to a certain protection domain kind. Each protection domain kind can have several protection domains.

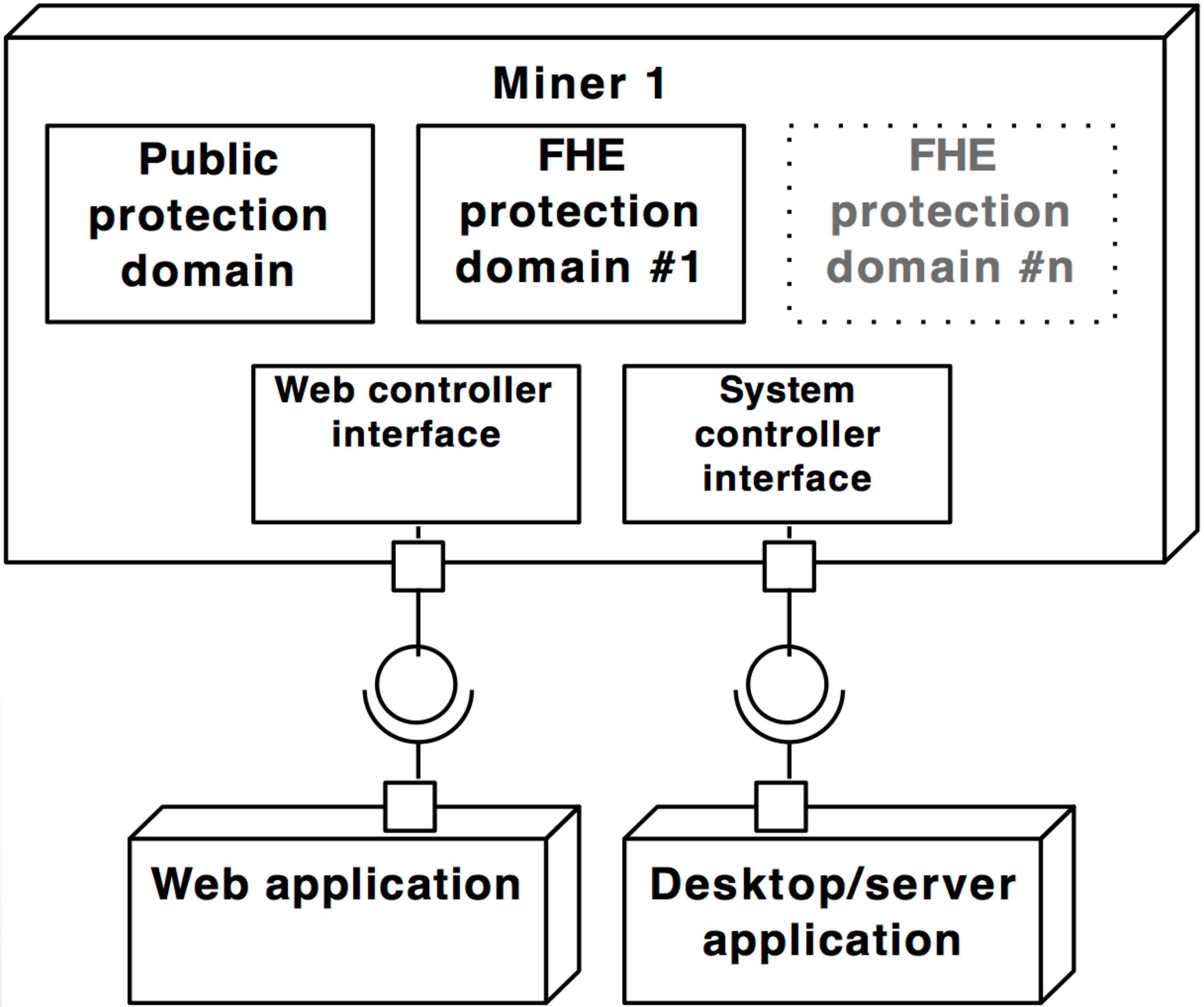
Examples of PD-s

- An FHE system running under a single key is a protection domain.
- A single physical MPC instance is a protection domain.
- A public machine is a protection domain.

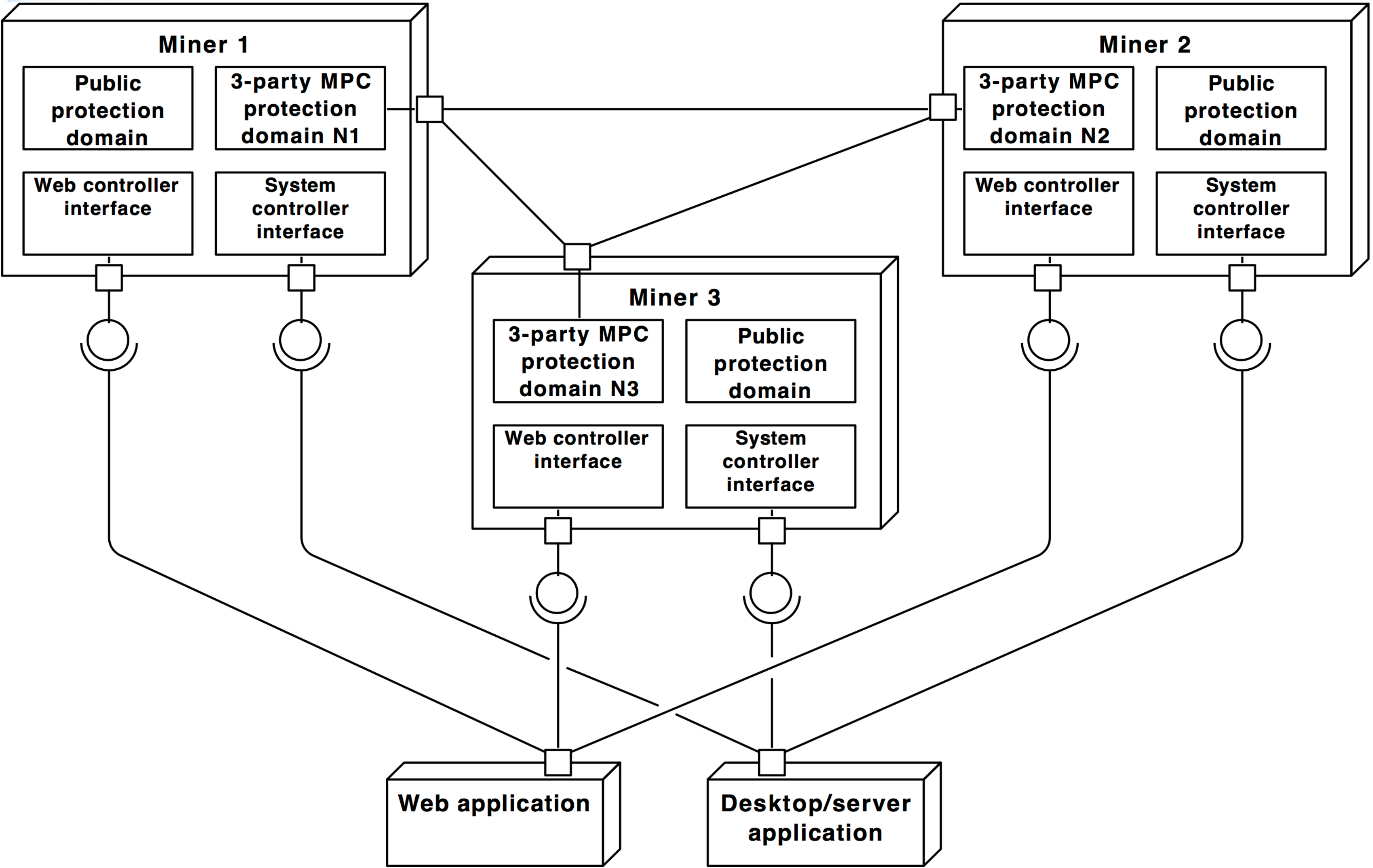
Examples of PD kinds

- A FHE system specified by its algorithms is a protection domain kind.
- A MPC system specified by its protocols is a protection domain kind.
- Public computation systems are a protection domain kind.

Single node with FHE PDs

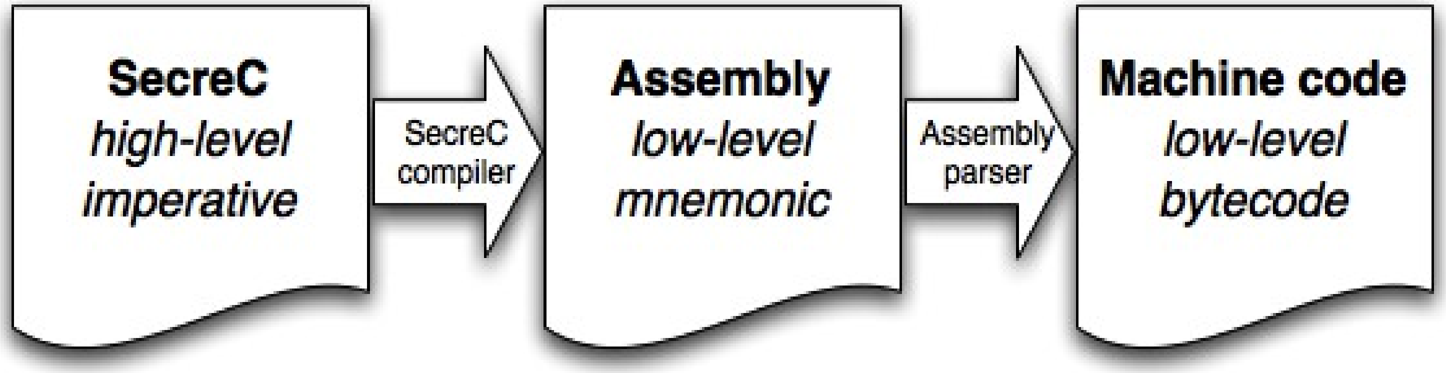


Three nodes with a MPC PD


Miner 1
Public protection domain
3-party MPC protection domain N1
Web controller interface
System controller interface
Miner 2
3-party MPC protection domain N2
Public protection domain
Web controller interface
System controller interface
Miner 3
3-party MPC protection domain N3
Public protection domain
Web controller interface
System controller interface
Web application
Desktop/server application

CYBERNETICA

Programming the machine



- The developer writes SecreC code.
- The resulting assembly code is deployed in the virtual machine.-
- The virtual machine parses it and stores it in memory as bytecode.

The low-level machine

- The basic machine is public.
- I.e, the program flow is not hidden.
- It has a stack and registers.
- Non-public PDs are used through system calls to their implementation.

SecreC 2 design goals

- SecreC is a high-level algorithm language for expressing algorithms that process confidential data.
- It has a type system supporting protection domains.
- The language is designed for implementing data mining algorithms.

Example of SecreC 2 code

```
kind additive3p;              // declare PD kind
domain private additive3p;    // instance of the PD

void main () {                // main function
  private int a, b, c;        // private data
  a = b + c;                  // private computation
  public int d;               // public data
  d = declassify (a);         // make private public
  publish (d);                // send to client
}
```

Writing library functions

```
template<domain T1, domain T2, domain T3>
T1 int [[1]] operator* (T2 int[[1]] x, T3 int[[1]] y)
{
  T1 int [[1]] result (size(x));
  public int i;
  assert(size(x) == size(y));
  for (i = 0; i < size(x); i++) {
    result[i] = x[i] * y[i];
  }
  return result;
}
```

PD specialization

```
template<domain T1:additive3p>
T1 int [[1]] operator* (T1 int[[1]] x, T1 int[[1]] y)
{
  // make a system call to a special function
  // implemented within the protection domain,
  // passing x and y as parameters
}
```

- Useful, when a PD implementation has a „hardware-accelerated“ implementation of a certain primitive.

Future work

- Sharemind 3 and SecreC 2 will become usable in 2012.
- We are interested in collaborating to implement new protection domains and protocol suites.
- We are looking for interesting applications and people interested in developing them.