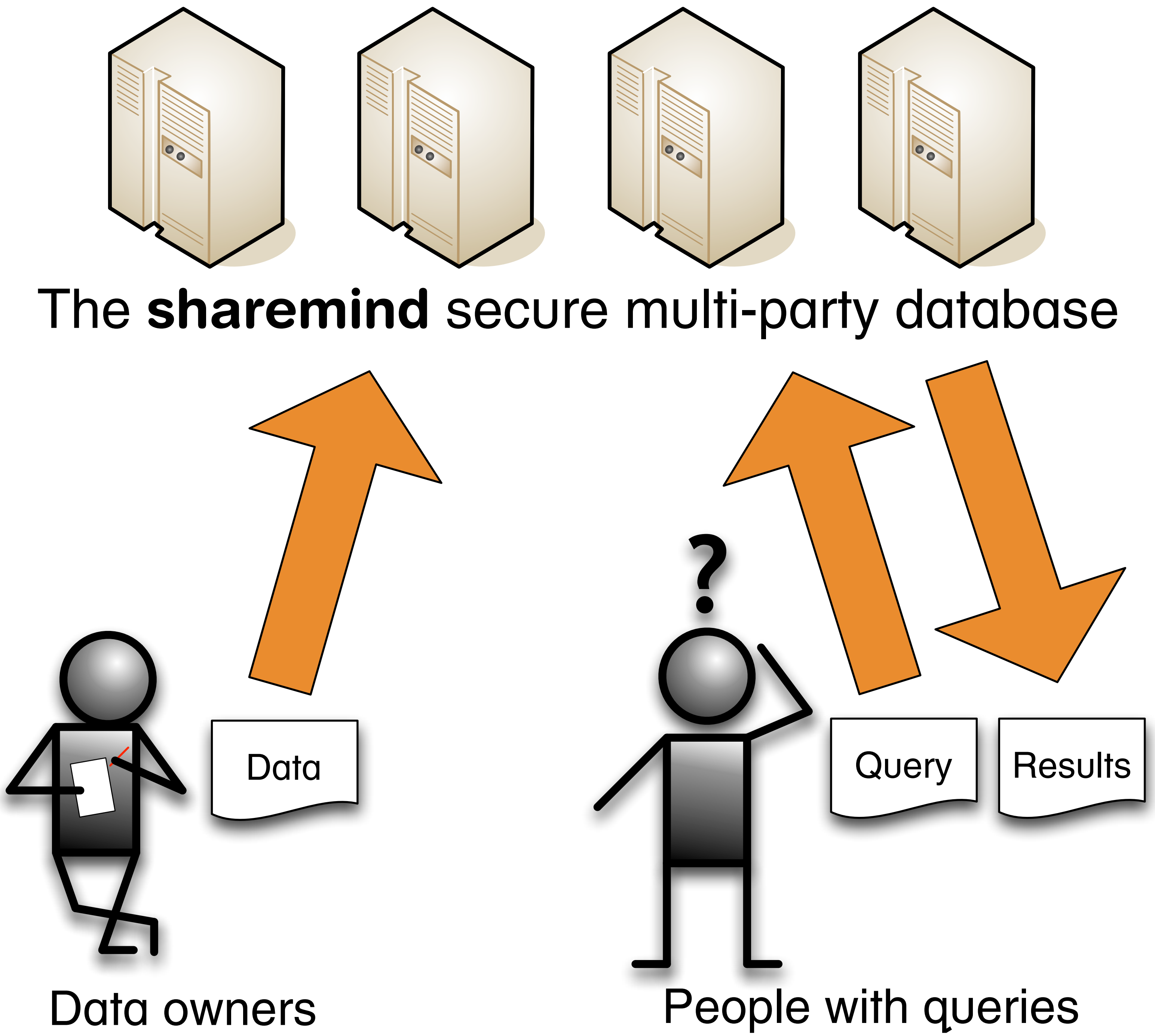
Easily programmable secure multi-party computation on integers, strings and floating point numbers

Dan Bogdanov
Sharemind project lead
dan@cyber.ee

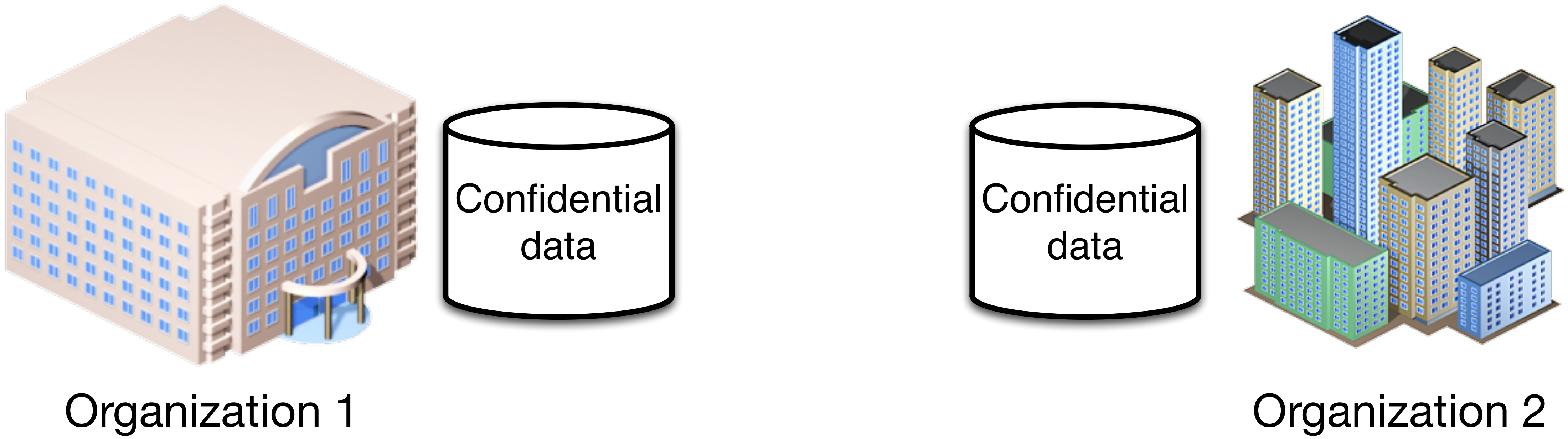
https://sharemind.cyber.ee/

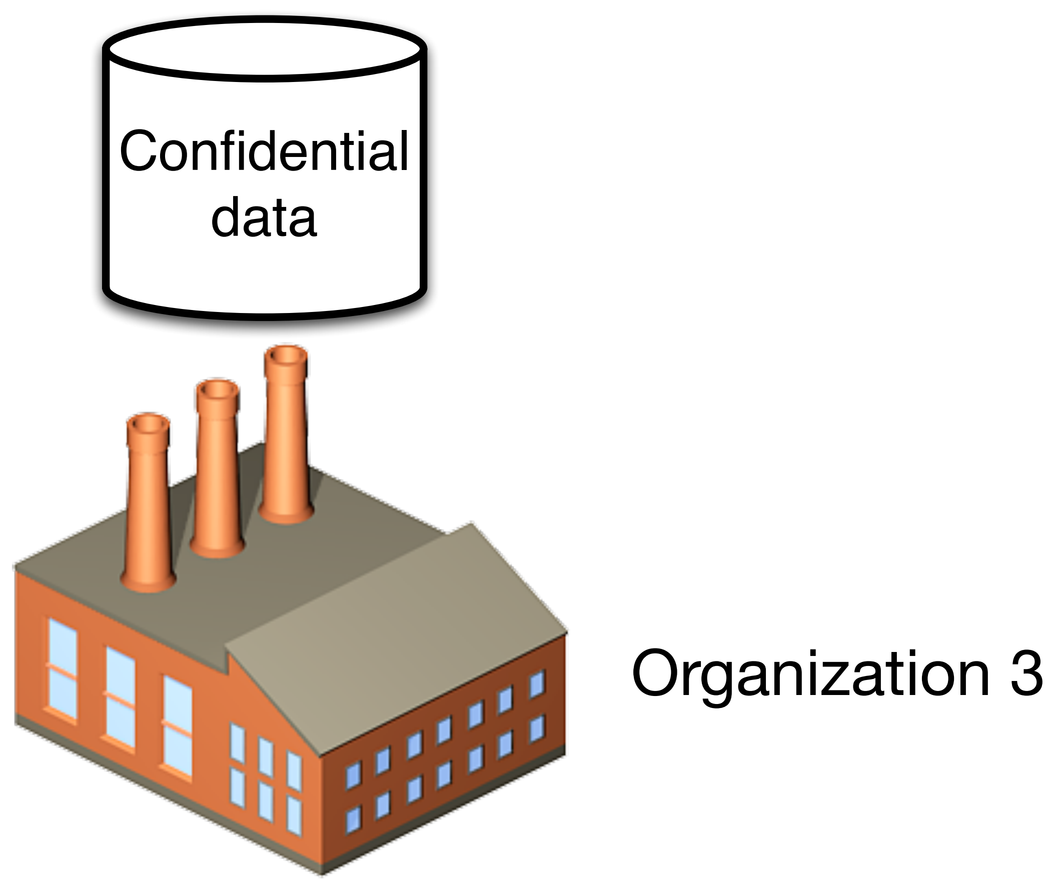
privacy preservation in statistics and data mining

Providing Security-as-a-Service



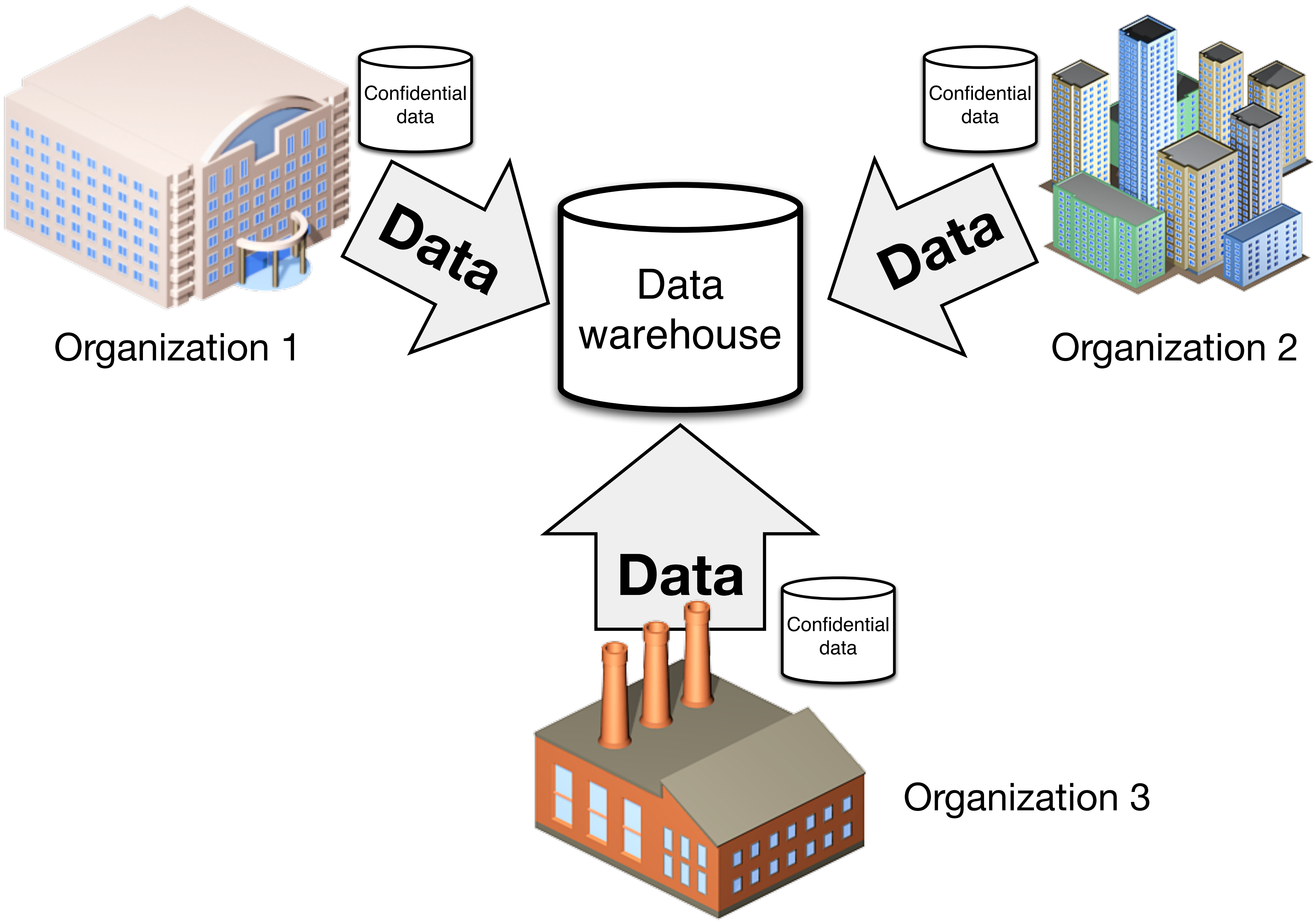
A typical problem statement



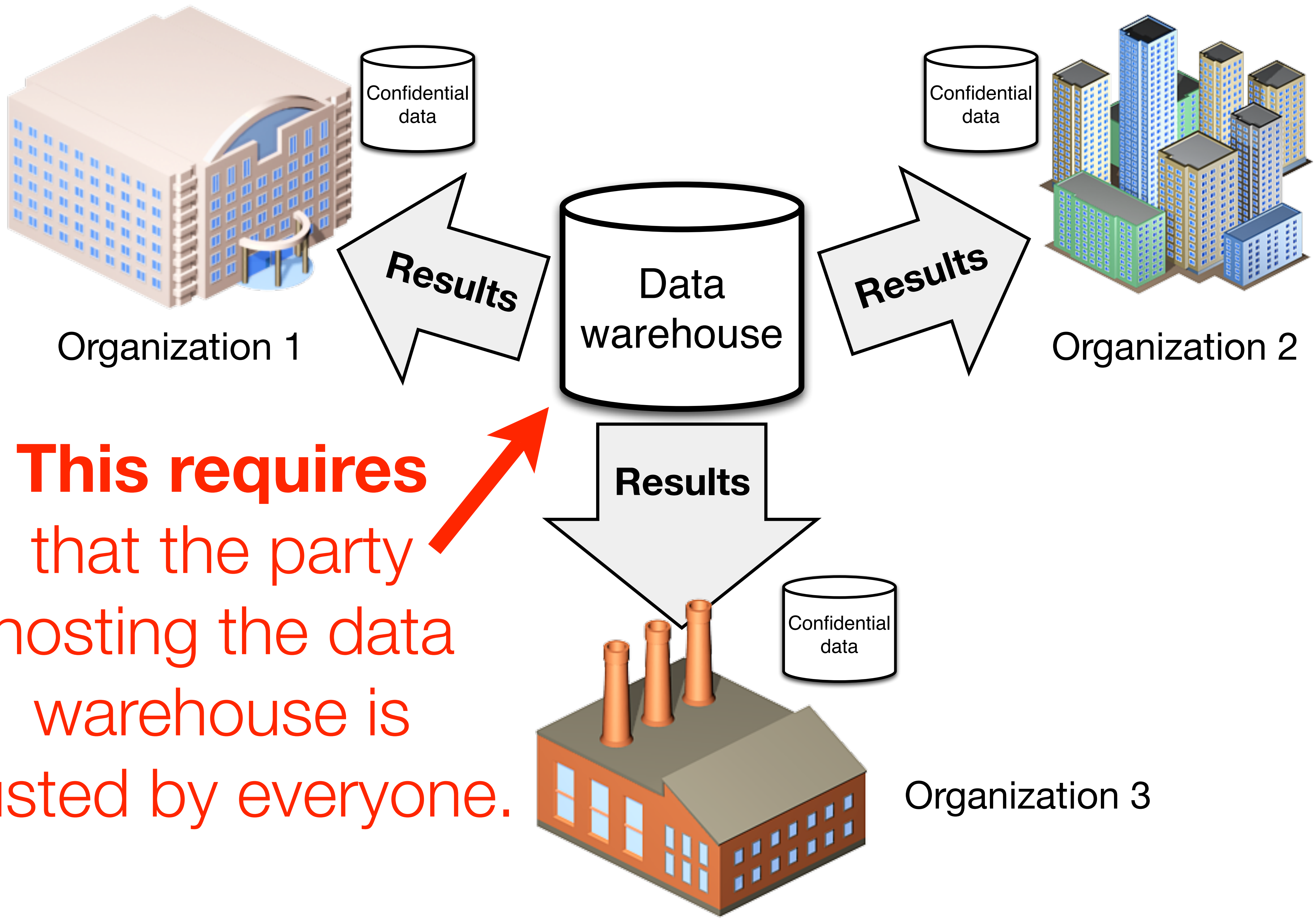


How do we
jointly analyze data
without showing
it to others?

A typical (but insecure) solution



A typical (but insecure) solution



This requires that the party hosting the data warehouse is trusted by everyone.

Our specific goal

- **Secure data aggregation**
  - Analyze data collected from several sources
  - Build services that package this technology.
- **Simple statistics and complex algorithms**
  - Compute sums and averages, use filtering.
  - Perform complex analyses like market basket analysis, clustering, regression and so on.

Security measures and guarantees

- **The data entry application protects input data.**
  - Only the data owner sees the input values.
- **The database of each server leaks no data.**
  - Defense against insiders (e.g. system administrators).
  - Some degree of protection against malicious hacking.
- **The servers run only agreed-to computations.**
  - Protection against malicious queries.

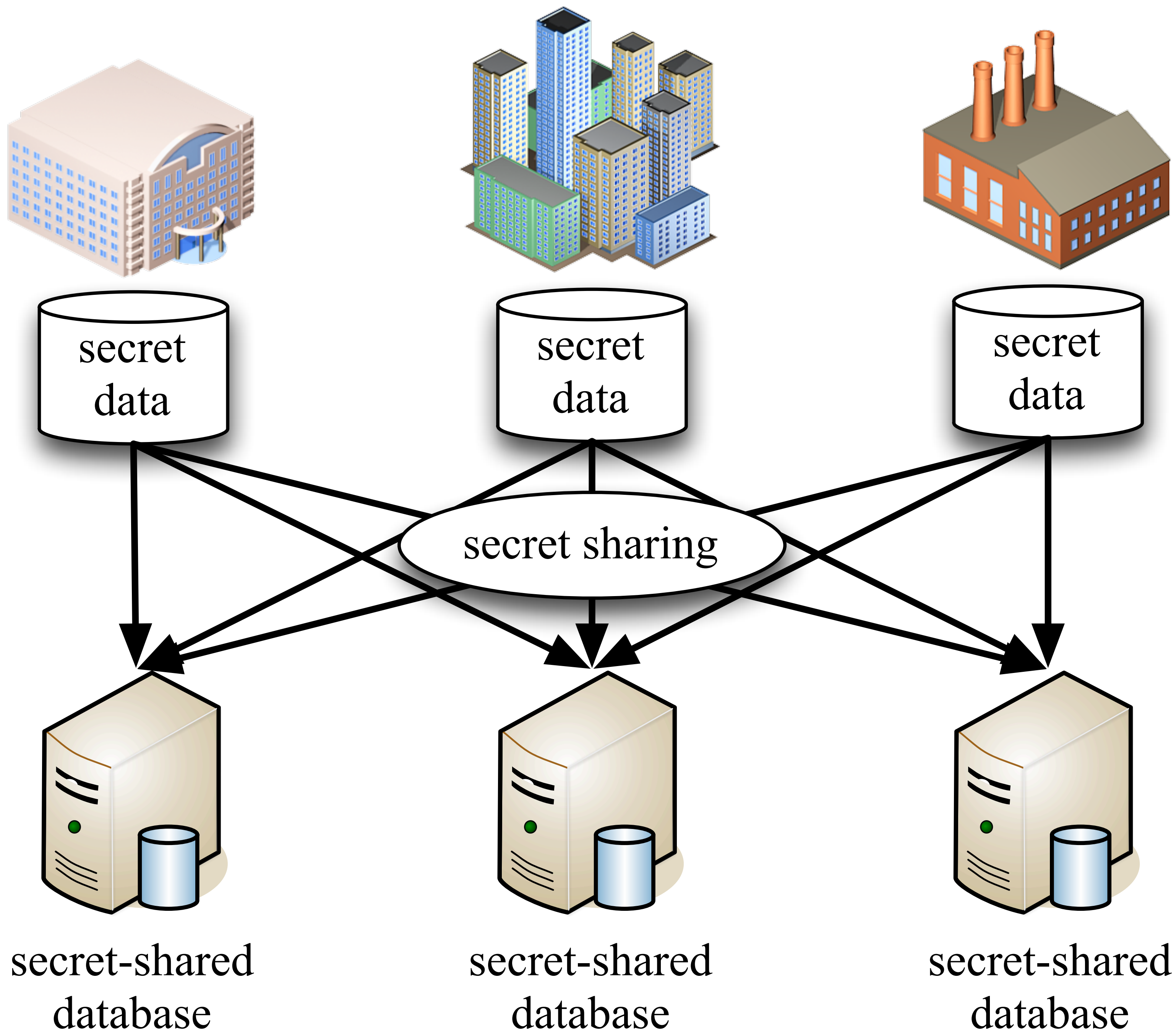
overview of sharemind 2

Secure computation à la sharemind

- We use additive secret sharing on 32-bit unsigned integers [BLW08].
- Both public and private values are from $\mathbb{Z}_{2^{32}}$.
- Three miner servers store the data and perform secure multi-party computation.
- Any number of controller applications provide data and request computations.
- Ideally, we can show information-theoretic security.

[BLW08] Bogdanov, Dan., Laur, Sven., Willemson, Jan. **Sharemind: a framework for fast privacy-preserving computations**.
In Proceedings of 13th European Symposium on Research in Computer Security, ESORICS 2008, LNCS, vol. 5283, pp. 192-206. Springer, Heidelberg (2008)

Getting data into sharemind

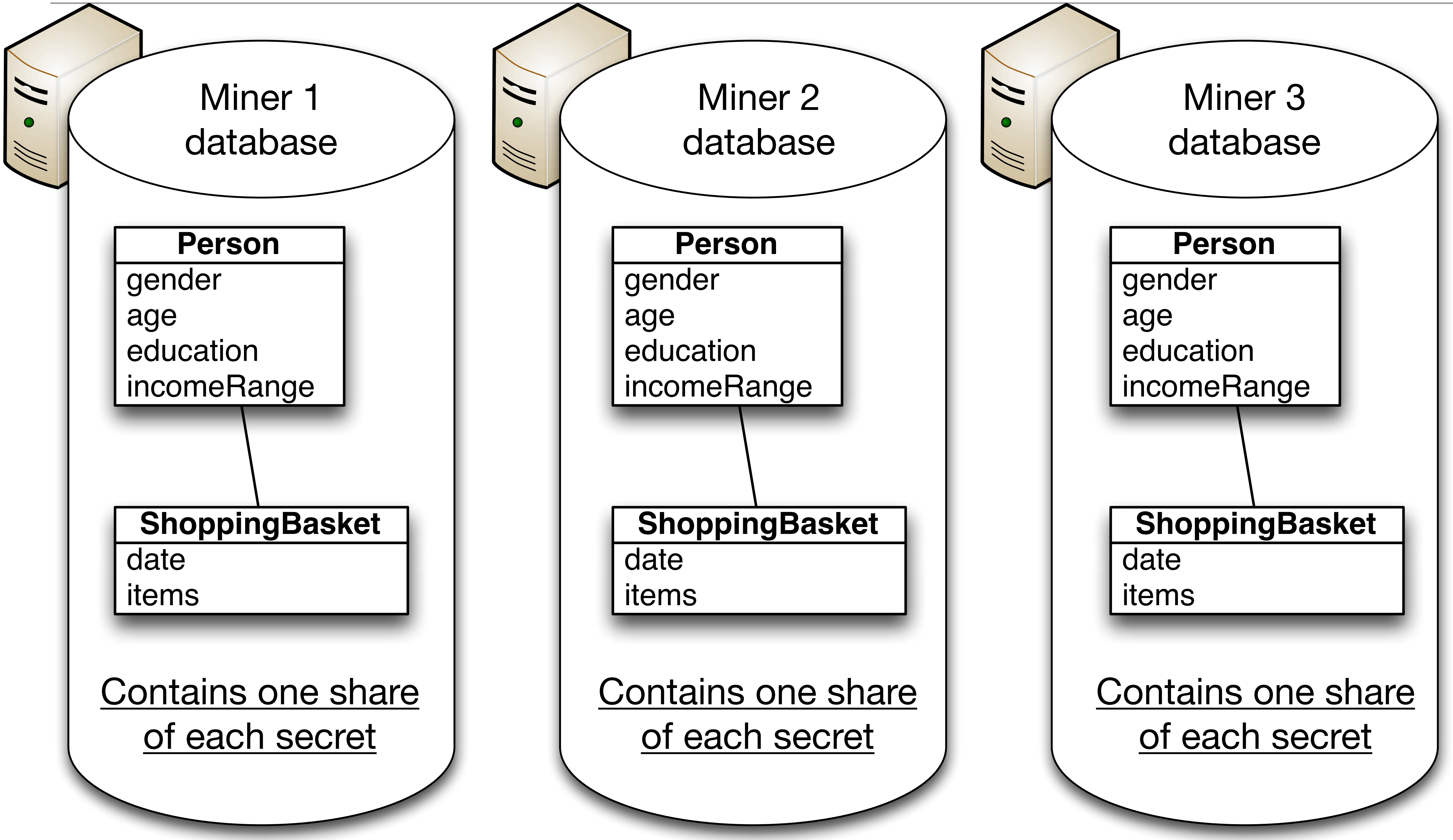


Features of controller applications

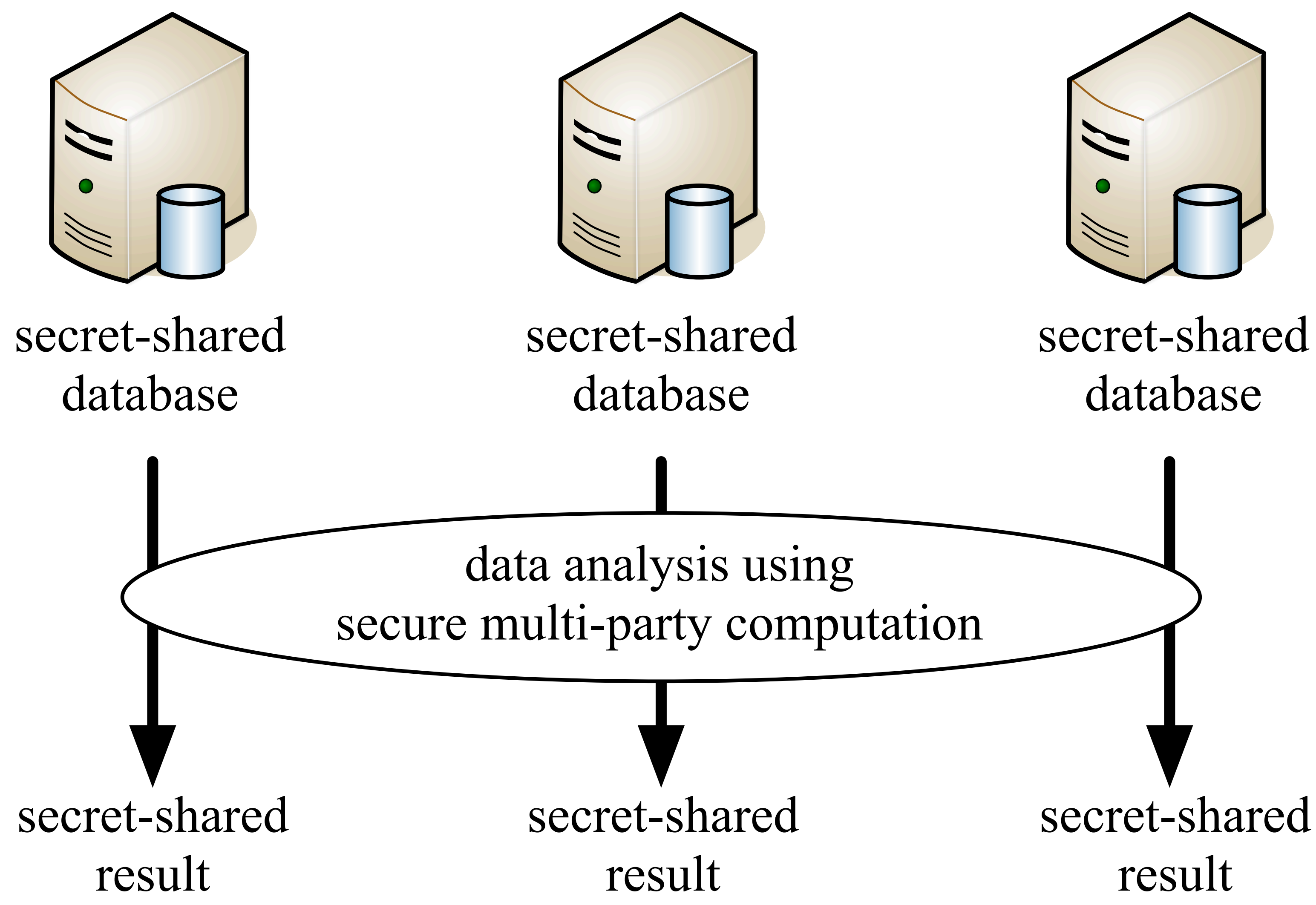
- Controller applications are built using the *controller library.*
- Different controller libraries exist for desktop and web applications [TB09].
- Mobile versions of the controller library are planned.
- The controller application automatically handles secret sharing when data is entered and when results are received.

[TB09] Talviste, Riivo., Bogdanov, Dan. **An improved method for privacy-preserving web-based data collection**. Cybernetica research report T-4-5. 2009. Available at: http://research.cyber.ee/

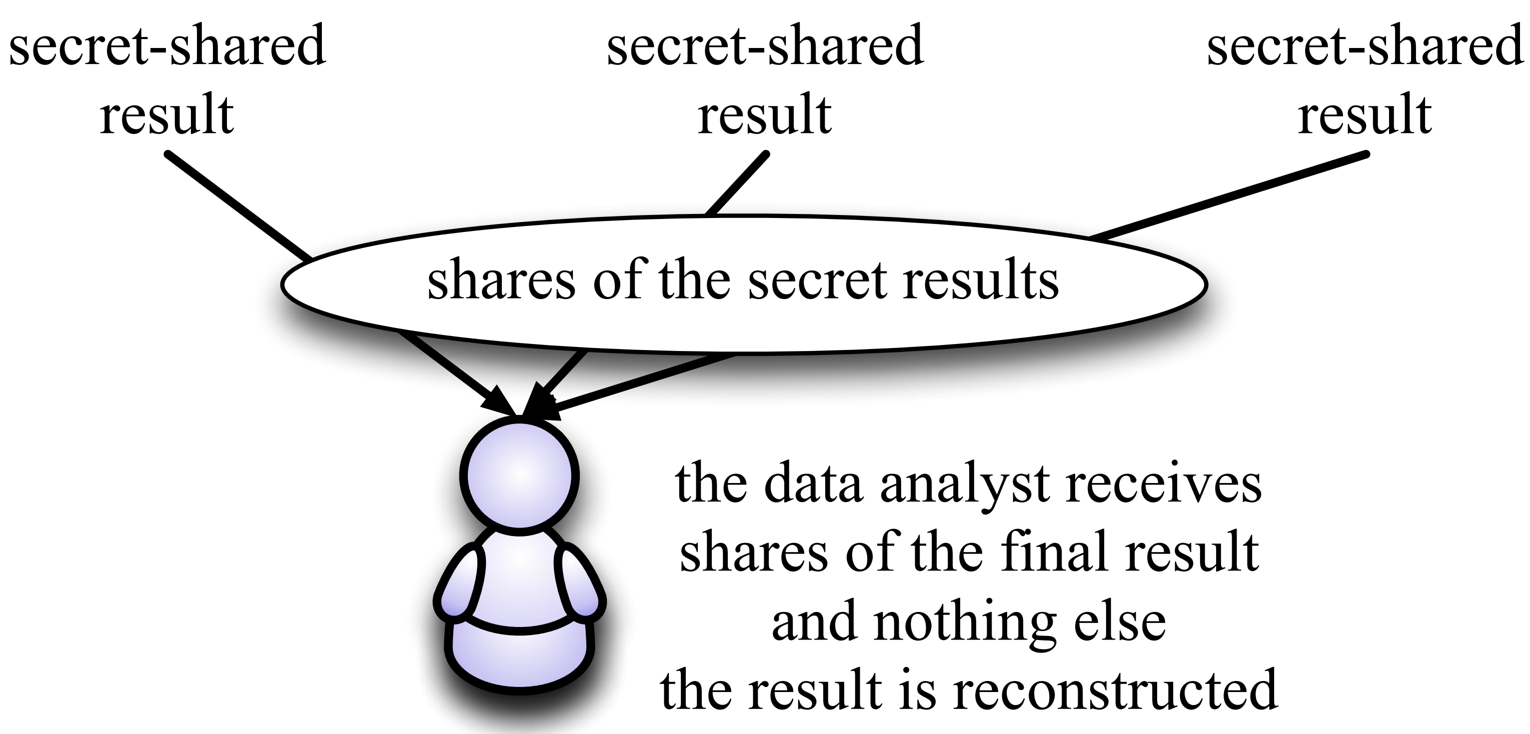
The sharemind built-in database



Processing data on sharemind



sharemind controls result publishing



Secure operations on sharemind

- Additive secret sharing is additively homomorphic so we get addition and multiplication by constant for free.
- We use custom protocols for all other operations.
- We have security and correctness proofs for these protocols together with universal composability proofs that allow them to be used in a programmable system.
- The current protocol suite is not yet published [BNTW].

[BTNW] Bogdanov, Dan., Niitsoo, Margus. Toft, Tomas.,Willemson, Jan. **High-performance secure multi-party computation for data mining applications.** Unpublished.

Performance in lab conditions (LAN)

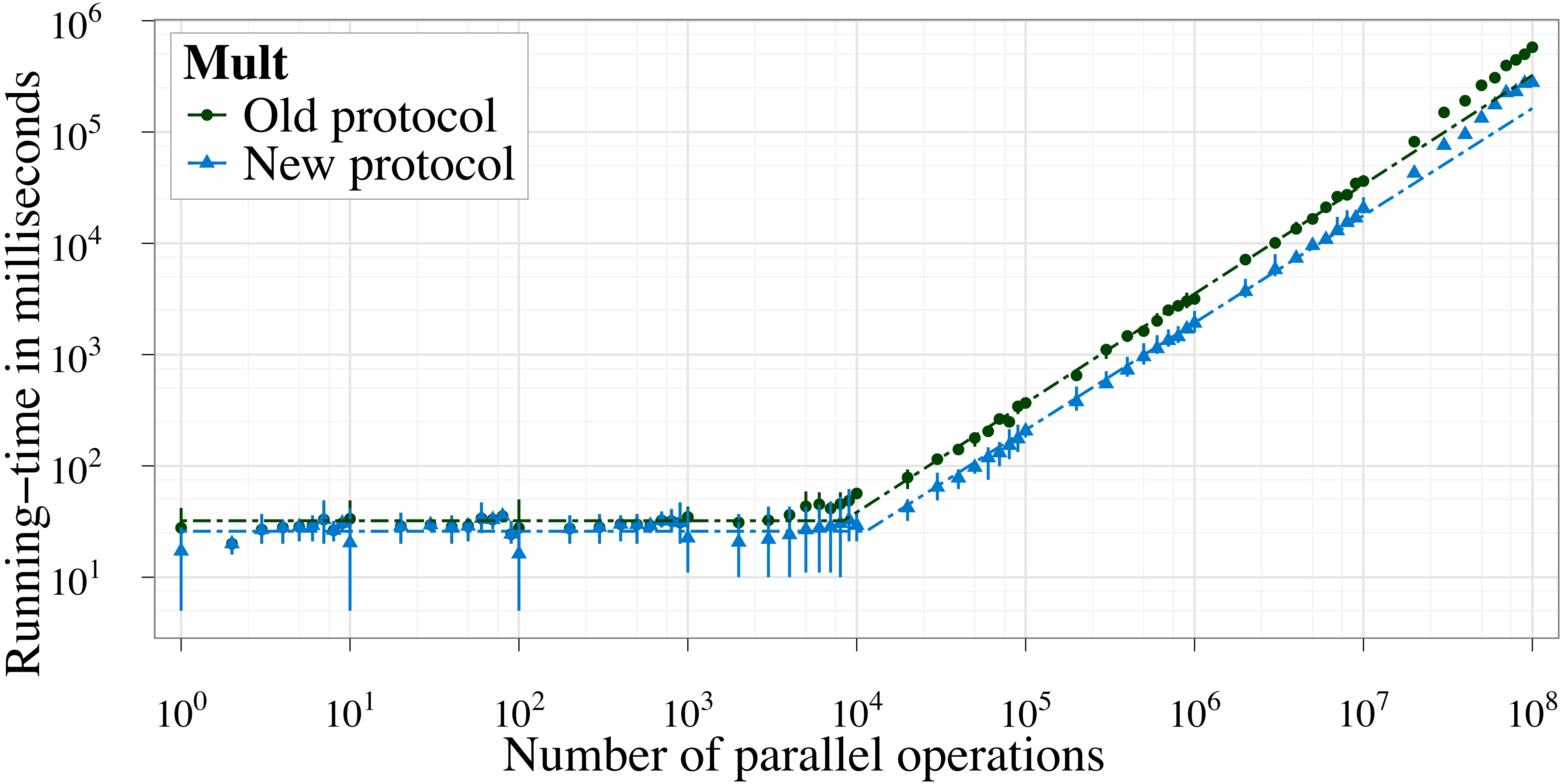
Protocol	Rounds	SISD	SIMD	SIMD Hz
Addition	local operation	-	0,015 µs	**66 MHz**
Multiplication w public	local operation	-	0,006 µs	**166 MHz**
Cast bool to int	1	15,3 ms	0,8 µs	**1,25 MHz**
Multiplication w private	2	25,9 ms	1,8 µs	**555 KHz**
Equality	$l + 2$	101 ms	5,0 µs	**200 KHz**
Greater-than	$l + 3$	113 ms	51 µs	**20 KHz**
Bit decomposition	$l + 3$	122 ms	15,7 µs	**64 KHz**
Division w public	$l + 4$	124 ms	44 µs	**23 KHz**
Division w private	$4l + 9$	390 ms	534 µs	**1,9 KHz**

Note: All operations are on 32-bit unsigned integers.

Note: $l = \log_2(numberOfBitsInDataType)$

[BTNW] Bogdanov, Dan., Niitsoo, Margus. Toft, Tomas.,Willemson, Jan. **High-performance secure multi-party computation for data mining applications.** Unpublished.

Saturation points in performance



[BTNW] Bogdanov, Dan., Niitsoo, Margus. Toft, Tomas.,Willemson, Jan. **High-performance secure multi-party computation for data mining applications.** Unpublished.

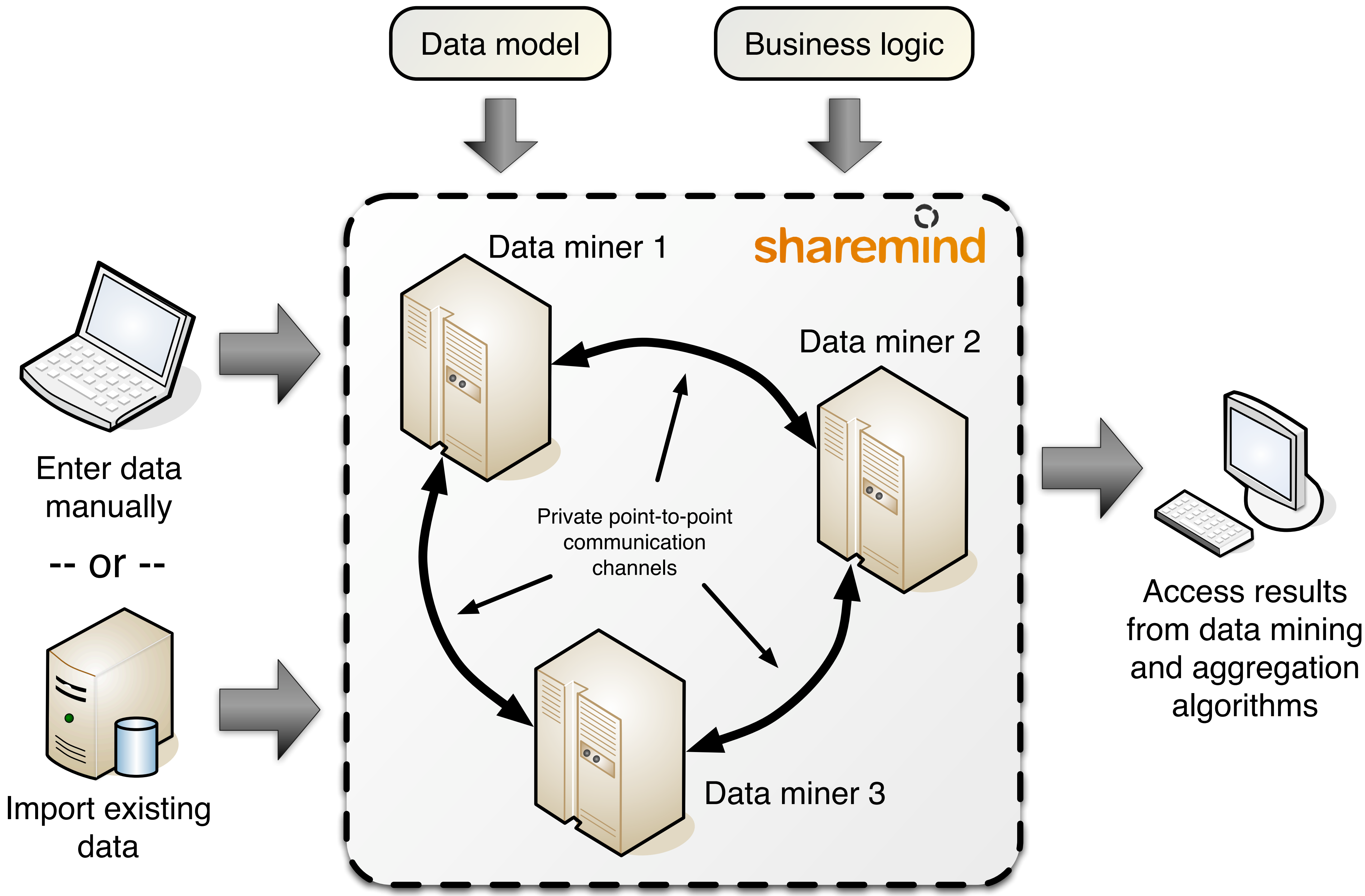
Performance on an international cloud

- We deployed Sharemind internationally, with miners in:
  - United States (West coast)
  - United Kingdom (London)
  - Japan (Tokyo)

Protocol	SIMD (100 000 parallel ops)
Cast bool to int	18 µs per operation
Multiplication w private	36 µs per operation
Equality	78 µs per operation
Greater-than	380 µs per operation
Bit decomposition	1,58 ms per operation

tools for creating secure applications

Deployment of a sharemind system



Programming secure computations

- The secure functionality is programmable in an assembly language that is interpreted by Sharemind.
- Internally, Sharemind has a private stack and public and private registers to support the implementation of algorithms.
- All registers store vectors to better support SIMD operations.
- The design is described in [BL10].

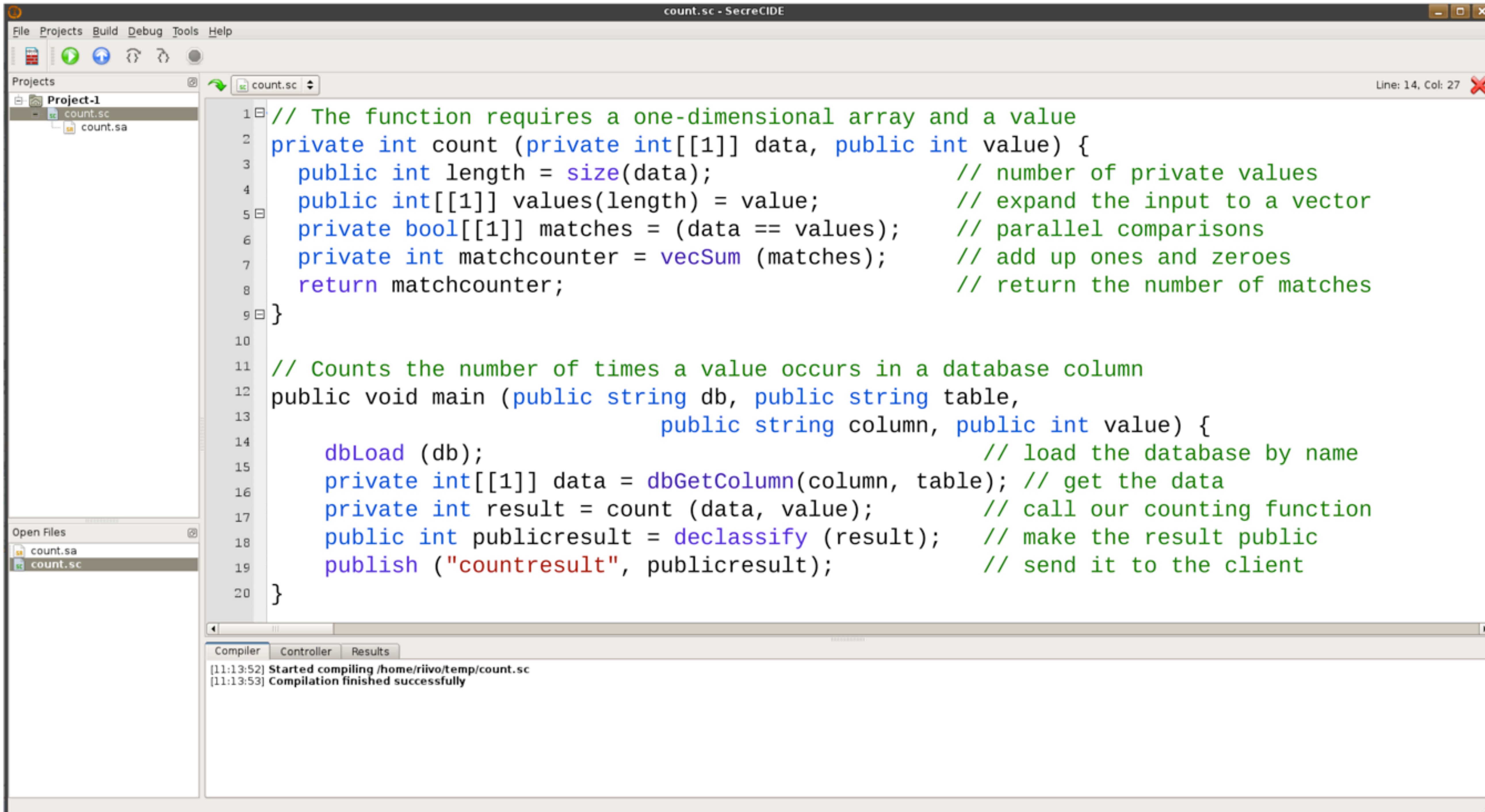
[BL10] Bogdanov, Dan; Laur, Sven. **The design of a privacy-preserving distributed virtual machine**.
In the Collection of AEOLUS theoretical findings. Deliverable D1.0.6. AEOLUS project IP-FP6-015964. 2010.

The SecreC language

```
public int count (private int[[1]] data,
                  public int value)
{
  public int length = size (data);
  private int matchcounter = 0;
  public int i = 0;
  for (i = 0; i < length; i++) {
    private bool match = (data[i] == needle);
    matchcounter += match;
  }
  return declassify (matchcounter);
}
```


[J10] Jagomägis, Roman. **SecreC: a Privacy-Aware Programming Language with Applications in Data Mining**. Master's thesis. University of Tartu, 2010.
[R10] Ristioja, Jaak. **An analysis framework for an imperative privacy-preserving programming language**. Master's thesis. University of Tartu, 2010.

The SecreCIDE developer tool



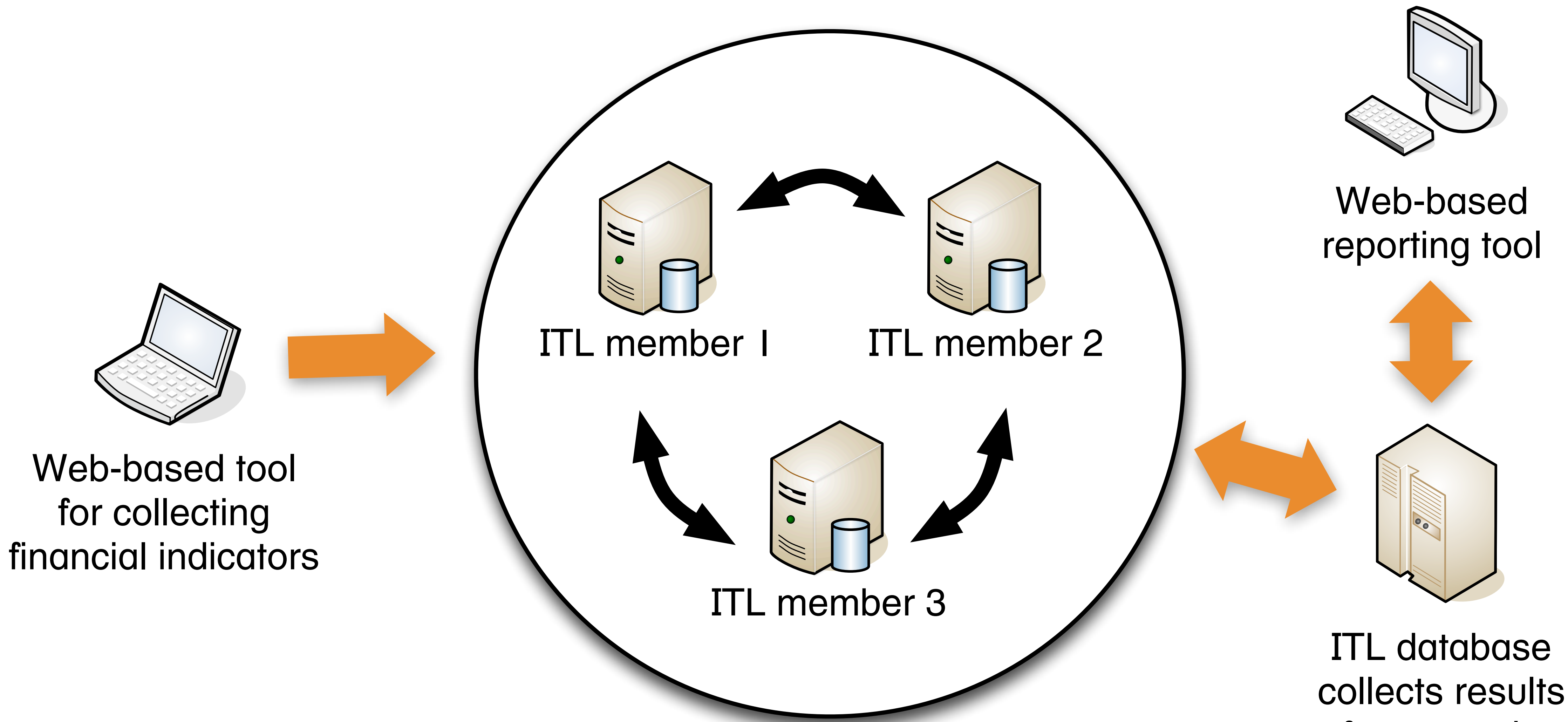
[RR10] Rebane, Reimo. **An integrated development environment for the SecreC programming language**. Bachelor's thesis. University of Tartu, 2010.

The sharemind SDK is freely available

- Sharemind SDK version 2012.04 is the latest version.
- It contains:
  - a developer version of the Sharemind 2.1 machine,
  - a compiler for the SecreC programming language,
  - a controller library for C++ applications,
  - example SecreC code and applications
- See https://sharemind.cyber.ee/ for downloads.

applications

sharemind in financial data analysis



A Sharemind installation deployed by
three independent members of the
ITL consortium performs secure computations

[BTW12] Bogdanov, Dan., Talviste, Riivo. Willemson, Jan. **Deploying secure multi-party computation for financial data analysis (Short Paper)**. In Proceedings of the Sixteenth International Conference on Financial Cryptography and Data Security 2012. To appear.

Secure computation algorithms used

- The analyses were implemented in SecreC.

Analysis operation	**Applied secure computation primitives**
Oblivious filtering to process only values that were entered by the user.	Boolean values, integer values, casting booleans to integers, multiplication.
Sorting individual data vectors	Oblivious array sorting using a sorting network. Requires addition, multiplication, and comparison.
Calculating a composite indicator *added value per employee*	Division of secret values
Time series for financial indicators	Oblivious matrix sorting by a key column

[BTW12] Bogdanov, Dan., Talviste, Riivo. Willemson, Jan. **Deploying secure multi-party computation for financial data analysis (Short Paper)**. In Proceedings of the Sixteenth International Conference on Financial Cryptography and Data Security 2012. To appear.

[LZW11] Laur, Sven., Zhang, Bingsheng., Willemson, Jan. **Round-efficient Oblivious Database Manipulation.** In Proceedings of the 14th International Conference on Information Security, ISC 2011, LNCS, vol. 7001, pp. 262-277. Springer, Heidelberg (2011)

Frequent itemset mining

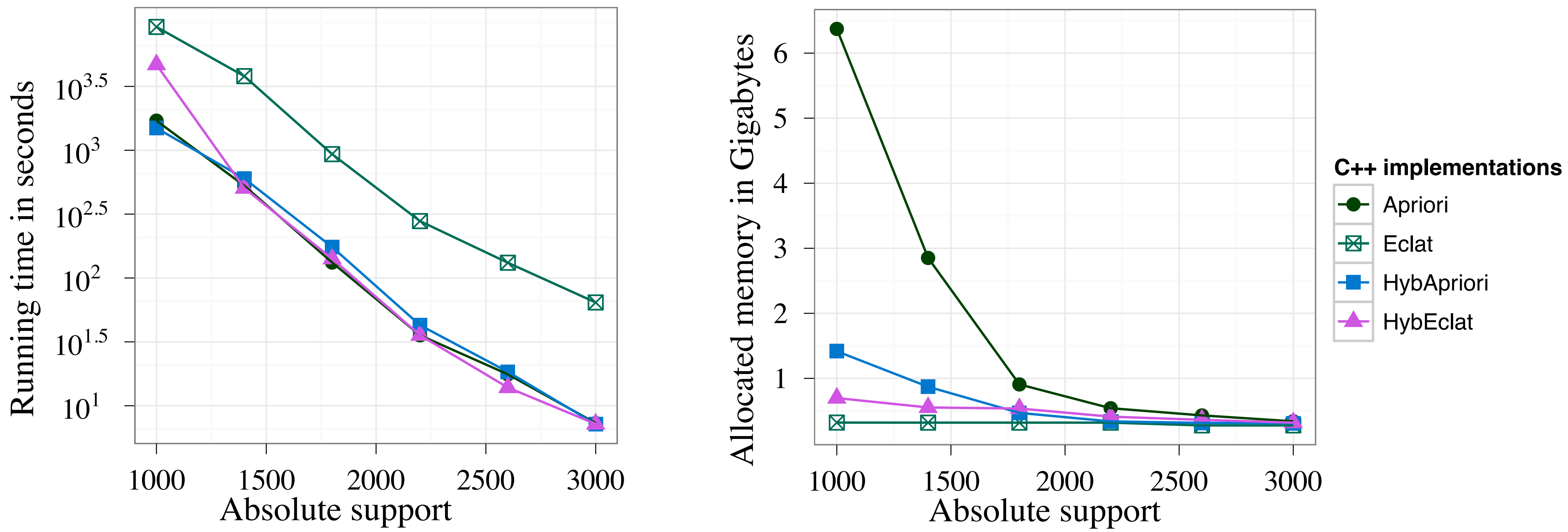
- FIM and association mining are used in problems like collaborative filtering and shopping basket analysis.
- We implemented four frequent itemset mining algorithms on Sharemind (Apriori, Eclat and hybrids).
- We benchmarked the result on three datasets.

Dataset	Transactions	Items	Density
mushroom	8124	119	19,3%
chess	3196	75	49,3%
retail	88163	16470	0,06%

[BJL12] Bogdanov, Dan., Jagomägis, Roman. Laur, Sven **A universal toolkit for cryptographically secure privacy-preserving data mining**. In Proceedings of the Pacific Asia Workshop on Intelligence and Security Informatics 2012. To appear.

sharemind processing large data

- To find frequent 11-item sets with support 2000 from the mushroom dataset Apriori needs to perform:
  - 71 548 068 secure multiplications
  - 8 926 secure greater-than comparisons



[BJL12] Bogdanov, Dan., Jagomägis, Roman. Laur, Sven **A universal toolkit for cryptographically secure privacy-preserving data mining**. In Proceedings of the Pacific Asia Workshop on Intelligence and Security Informatics 2012.

introducing
sharemind 3

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.

Adding more integer types

- The 32-bit integer type was hardcoded in Sharemind 2.
- However, our arithmetic protocols work in $\mathbb{Z}_{2^{2^n}}$.
- Therefore, we can let the applications use booleans, 8-bit integers, 16-bit integers and 64-bit integers too.
- Smaller integers are more efficient since they use less communication and storage space.
- Currently, we have used unsigned integers, but we could adapt some of the protocols to signed integers.

Moving from integers to real numbers

- Given that we now have secret values with different bit depths, we can consider more complex data types.
- Our first target is to suppert floating point numbers.
- The standard IEEE 754 32-bit floating point number looks like this:

- In Sharemind 3, we will use a modified version:

[WK12] Willemson, Jan., Kamm, Liina. **Secure Multiparty Floating Point Arithmetic.** Submitted.

Operations on floating point numbers

- We created data-independent circuits for adding and multiplying secret-shared floating point numbers.
- We do not process Not-a-Number cases. The protocols will not leak anything, but the result is undefined.
- After implementing addition and multiplication, we used Taylor series to compute basic functions such as sine, natural logarithm and square root.

[WK12] Willemson, Jan., Kamm, Liina. **Secure Multiparty Floating Point Arithmetic.** Submitted.

Benchmarks on floating point ops

- An unoptimized addition takes 1.4 seconds.
- An unoptimized multiplication takes 0.4 seconds.
- Sine computation (5 elements) takes 8.8 seconds.
- Natural logarithm (6 elements) takes 13.5 seconds.
- Square root (5 elements) takes 10.1 seconds.
- All operations will be more efficient if parallelized.

[WK12] Willemson, Jan., Kamm, Liina. **Secure Multiparty Floating Point Arithmetic.** Submitted.

Private string operations

- Given that we have different bit depth integers, we can use their arrays to implement ASCII and UTF strings.
- We can have two options for strings:
  - fixed length - potentially hides message length
  - variable length - string is as long as the array allocated for its storage
- Algorithms are slightly different for both cases.

Challenges for string algorithms

- Character manipulation, especially with UTF strings, requires bit-level operations to be efficient.
- However, if we use additive secret sharing, we need to use an expensive bit decomposition to manipulate bits.
- Therefore, for strings we will consider using XOR instead of addition in the secret sharing scheme.
- The same approach is useful also for other operations that need bit-level access to data - like secure AES.

[LTW12] Laur, Sven., Talviste, Riivo., Willemson, Jan. **AES block cipher implementation and secure database join on the Sharemind secure multi-party computation framework**. Submitted.

Challenges for string algorithms

- Standard string algorithms can make decisions based on the data and jump ahead (eg. in searching).
- To guarantee data independence, we often need to run naïve versions of algorithms and brute force searches
- However, brute force can typically be heavily parallelized with SIMD operations.
- The alternative is to leak some (possibly aggregated) bits about the string's contents, but such decisions should be driven by applications.

Adding support for new paradigms

- Sharemind 2 is limited to three parties and solutions based on secret sharing. The limits will be removed.
- However, there are interesting protocols and primitives out there that may work in the same application model:
  - Homomorphic encryption can help reduce the number of required servers and the communication.
  - Security in the consistent or malicious model will help protect against outages and hacking.
- We will be happy to implement interesting protocols.

Keeping it all easy for the developers

- We want all new data types to be available in the SecreC language for ease of use.
- We want the programming experience to be similar to existing techniques. Sharemind should be perceived similarly to a database and application server.
- Programs written in SecreC should be forward-compatible with new results in cryptography.
- The Sharemind machine is responsible for hiding the complexities of scheduling protocols.

Future work

- In the coming years we will be looking for novel practical protocol designs for inclusion in Sharemind.
- We want to extend the practical applicability of secure computation technology by building more prototypes.
- The technology can be used to solve real-life problems and we will jump to the opportunity of doing so.
- We will maintain a freely available toolkit for creating secure computation applications for academic use.
- We welcome all collaboration opportunities.

sharemind

https://sharemind.cyber.ee/

sharemind@cyber.ee