Security and Reliability in the Cloud

Pål Ellingsen

Bergen University College

Outline

1) Intro to Cloud Computing
2) Cloud Computing Security
    1) Security issues if you trust the provider
    2) What to do if you don't trust the provider?
    3) Homomorphic Encryption

What is Cloud Computing?

- Provide computing services over a network
- Physical computing resources are not under client control
- Clear separation between provider and clients
- Abstraction of resources are presented to the users
- Resources are shared between users
  - Elasticity
  - Resource utilization
  - Pay-per-use

Cloud Computing Service Models

Several different types of services can be provided by cloud technologies.

- **IaaS (Infrastructure as a Service)**
  provides access to processing capacity on computers (real or virtual), data storage, networking features.

- **PaaS (Platform as as Service)**
  provides infrastructure as IaaS and also a computing platform on which client can develop and deploy applications .

- **SaaS (Software as as Service)**
  provides complete applications to the client through a cloud platform.

- **XaaS (Anything as a Service)**
  a trend is to provide several different services to clients, including the above, but also messaging, monitoring, communication etc.

Cloud Computing Services

There is also a natural categorization based on how data is handled by the cloud services.

- **Storage**
  Services that stores data for the client application, but does not access the data on their own.

- **Computation**
  Services that has access to the client's data and uses it for computational operations to deliver the service.

Cloud Computing Security

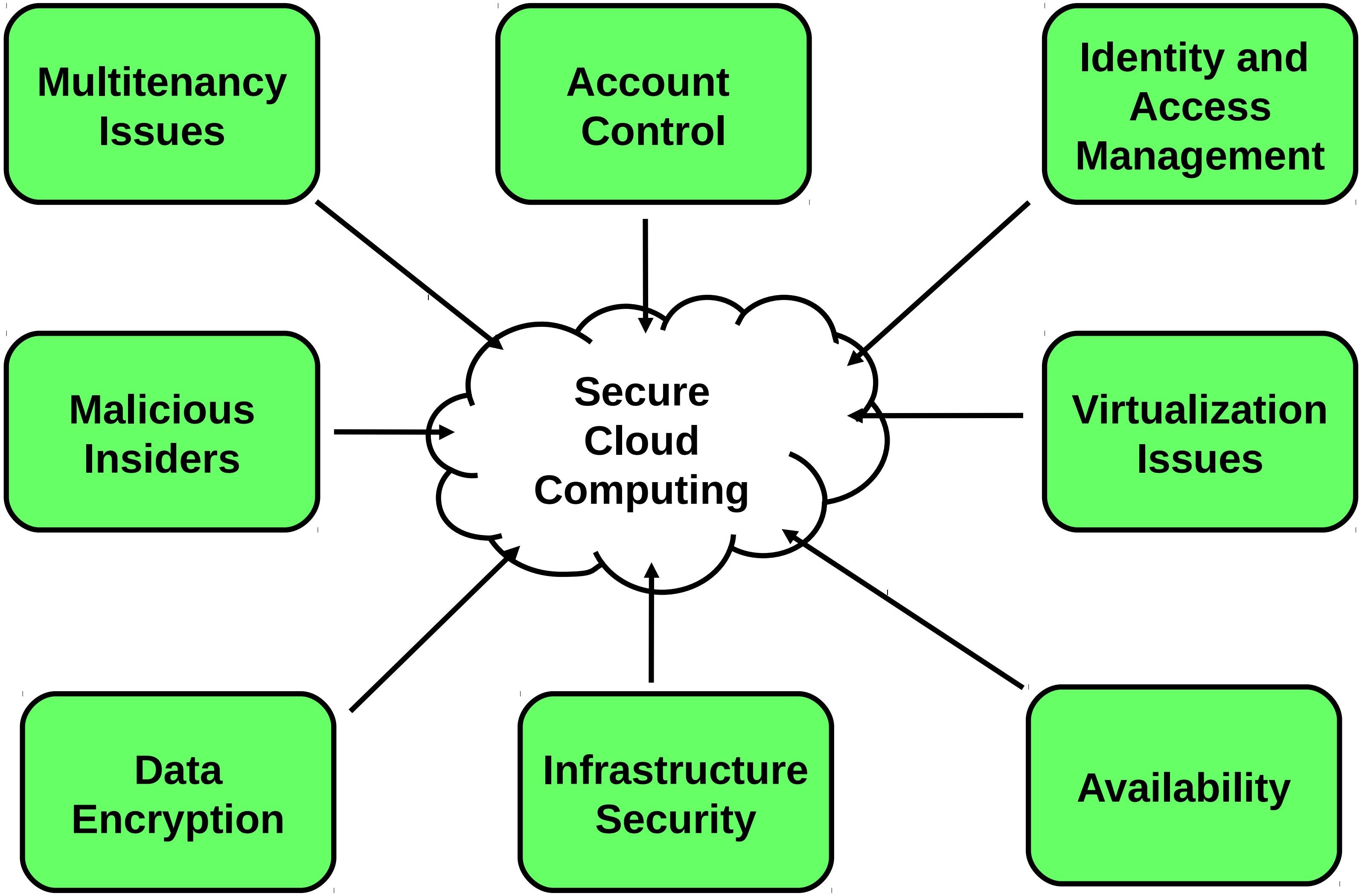
Cloud Computing Security

Corresponding to the two different ways information can be handled in a cloud computing system, there are two main questions that can be posed:

- **Can the provider be trusted?**
  When a client lets its data be handled by a provider, it must either trust the provider, or protect its data from being exposed to the provider.

- **Can the provider's infrastructure be trusted?**
  If a client decides to trust the provider, a different question is whether the provider's infrastructure can be trusted. It is normally the provider's responsibility to ensure this.

The Provider can be trusted



The Provider can be trusted

- **Security issues at application level.**

Service level	Users	Security requirements	Threats
Software as a Service (SaS)	End client applies to a person or organization who subscribes to a service offered by a cloud provider and is accountable for its use	• Privacy in multitenant environment • Data protection from exposure (remnants) • Access control • Communication protection • Software security • Service availability	• Interception • Modification of data at rest and in transit • Data interruption (deletion) • Privacy breach • Impersonation • Session hijacking • Traffic flow analysis • Exposure in network

Source: Zissis and Lekkas, 2012 [1]

The provider can be trusted

- **Security threats at virtual infrastructure level.**

Service level	Users	Security requirements	Threats
Platform as a Service (PaS) Infrastructure as a Service (IaS)	Developer–moderator applies to a person or organization that deploys software on a cloud infrastructure	• Access control • Application security • Data security, (data in transit, data at rest, remanence) • Cloud management control security • Secure images • Virtual cloud protection • Communication security	• Programming flaws • Software modification • Software interruption (deletion) • Impersonation • Session hijacking • Traffic flow analysis • Exposure in network • Defacement • Connection flooding • DDOS • Impersonation • Disrupting communications

Source: Zissis and Lekkas, 2012 [1]

The provider can be trusted

- **Security issues at physical level.**

Service level	Users	Security requirements	Threats
Physical datacenter	Owner applies to a person or organization that owns the infrastructure upon which clouds are deployed	• Legal not abusive use of cloud computing • Hardware security • Hardware reliability • Network protection • Network resources protection	• Network attacks • Connection flooding • DDOS • Hardware interruption • Hardware theft • Hardware modification • Misuse of infrastructure • Natural disasters

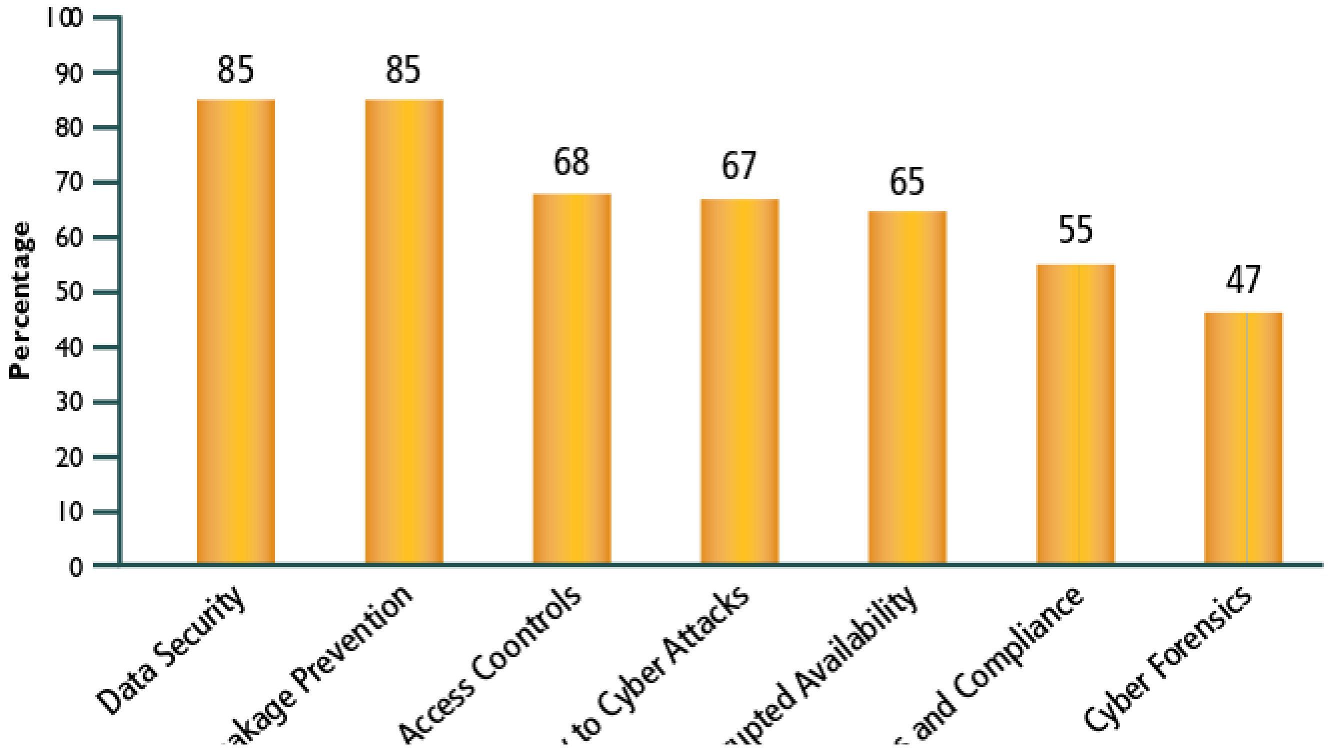
Source: Zissis and Lekkas, 2012 [1]

The provider can be trusted

To meet the security challenges mentioned above, there is a set of advantages of trusting a cloud computing service provider:

- **Advanced perimeter security**
- **Protection against DDoS attacks**
- **Data redundancy, fragmentation and dispersal**
- **General system redundancy and resiliency**
- **Professional security management**
- **Recovery services**
- **Advanced detection and logging services**
- **Incentive to invest in security**

Does it make sense to trust the provider?

- Cloud Security Alliance, *Cloud Computing Top Threats 2013:* **Data breaches is the #1 threat**
- 

Source: 8th annual Global Information Security Survey

Sensitive Data and Processing

There is a lot of data around that is very sensitive, but that also benefit from using the processing capability of the cloud.

- **Personal medical and biological data**
- **Business sensitive data (e.g. financial)**
- **Biometric data (e.g g for authentication)**
- **Electronic voting (statistics)**

In particular, the ability to make use of cloud processing capabilities for very large datasets is attractive
– Big Data in the Cloud.

The provider can't be trusted

- If the client decides not to trust the provider, data must be kept confidential from any part of the system that is not under control of the client.

- For the information itself, a simple solution is encrypting data before it is trusted to the provider.

    - Encrypting data will, however, prevent the provider from using the data e.g. for computation.
    - The client is restricted to storing the information, no processing.

The provider can't be trusted

- If the client decides not to trust the provider, data must be kept confidential from any part of the system that is not under control of the client.

- For the information itself, a simple solution is encrypting data before it is trusted to the provider.

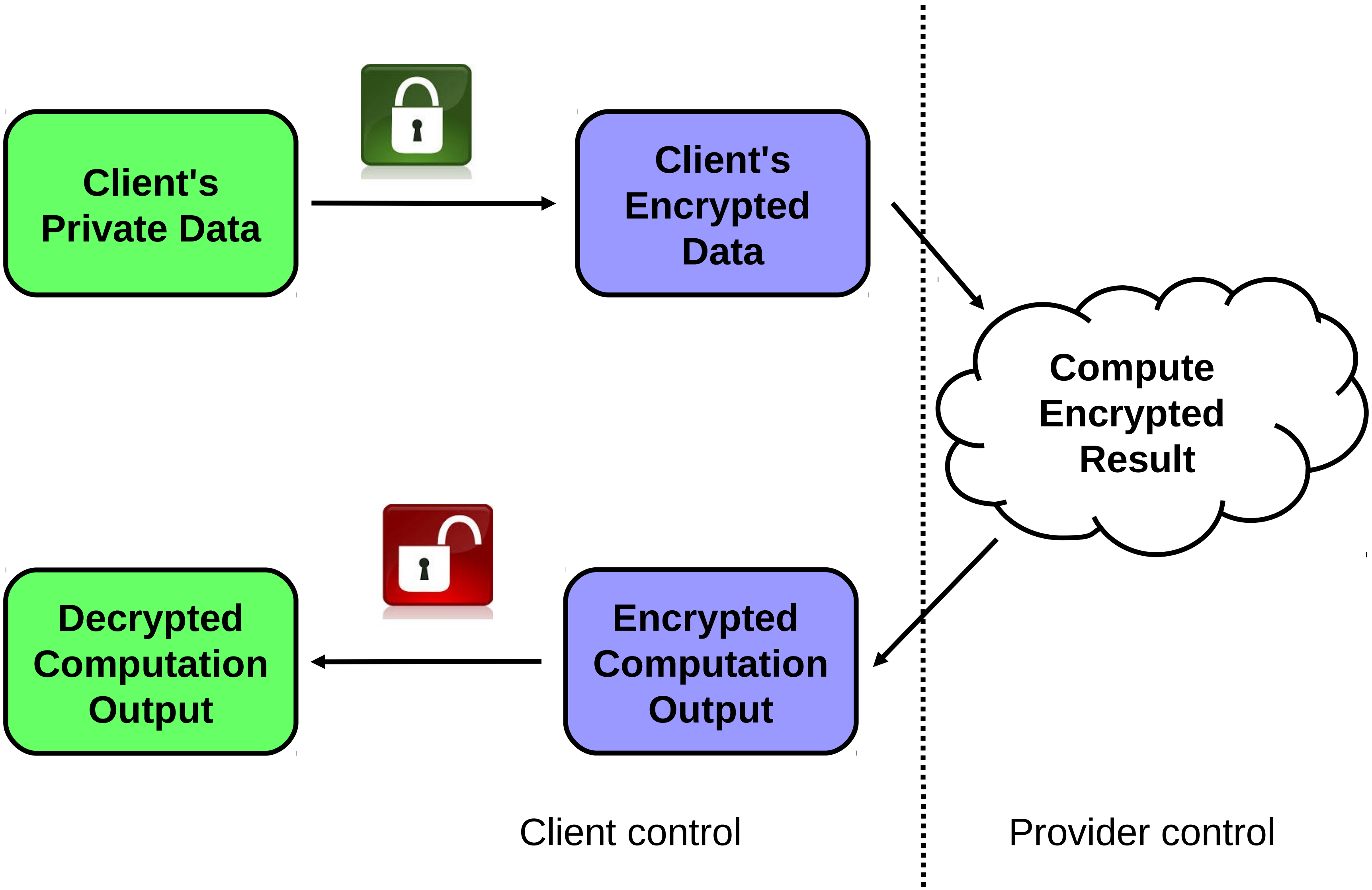
  - Encrypting data will, however, prevent the provider from using the data e.g. for computation.
  - The client is restricted to storing the information, no processing.

Homomorphic Encryption

A recent advance in cryptology opens up the possibility for letting the provider side do the computational work on such sensitive data without having access to the contents of the data:

- **Homomorphic encryption**
  Encryption schemes with the property that **computational operations may be performed on encrypted data.**

Homomorphic Encryption



Homomorphic Encryption

The description of an encryption as being

homomorphic

stems from the concept of

homomorphism

in mathematics:

- **Group Homomorphism**

A group homomorphism is a map $f: G \rightarrow H$ between two groups G and H with operations $*$ and $\circ$ such that the group operations are preserved: $f(g_1 * g_2) = f(g_1) \circ f(g_2)$ for all g_1, g_2 in G, where the product on the left-hand side is in G and on the right-hand side in H.

Homomorphic Encryption

Based on this, a more mathematical definition of homomorphic encryption is given by Yi et al. [5]:

Let (P, C, K, E, D) be an encryption scheme, where P, C are the plaintext and ciphertext spaces, K is the key space and E, D are the encryption and decryption algorithms.
Assume that the plaintexts forms a group $(P, *)$ and the ciphertexts forms a group $(C, \circ)$, then the *encryption* algorithm E is a map from the group P to the group C, i.e., $E_k : P \rightarrow C$, where $k \in K$ is either a secret key (in a symmetric key cryptosystem) or a public key (in a public-key cryptosystem).

For all a and b in P and k in K, if $E_k(a) \circ E_k(b) = E_k(a * b)$ the encryption scheme is said to be homomorphic.

Homomorphic Encryption

In the definition above, only one operation was considered. It is however possible to extend the definition to cover an arbitrary set of operations on the plaintext and ciphertext spaces. This gives rise to two groups of homomorphic cryptosystems.

- **Partially homomorphic cryptosystems**
  A limited number of operations can be applied to encrypted data.

- **Fully homomorphic cryptosystems**
  An arbitrary number of operations can be applied to encrypted data.

Homomorphic Encryption

- The underlying principle of homomorphic encryption has been known for almost 40 years.
- It turned out that the RSA cipher is a partly homomorphic cipher under multiplication.
- This discovery led the inventors of RSA to speculate if it was possible to design a cipher that was homomorphic under the application of an arbitrary number of **all** operations of the plaintext and ciphertext spaces.
- The principle was named *privacy homomorphism*.

Homomorphic Encryption Example

- In an unpadded RSA cipher, assume that the public key $p_k = (n, e)$, the plaintexts form a group $(P, \cdot)$, and the ciphertexts form a group $(C, \cdot)$, where $\cdot$ is the modular multiplication operation.

- For any two plaintexts m_1, m_2 in P, it holds that

$$E(m_1, p_k) \cdot E(m_2, p_k) = m_1^e \cdot m_2^e \pmod n$$

$$= (m_1 \cdot m_2)^e \pmod n = E(m_1 \cdot m_2, p_k)$$

- Thus, the unpadded RSA has the homomorphic property under multiplication.

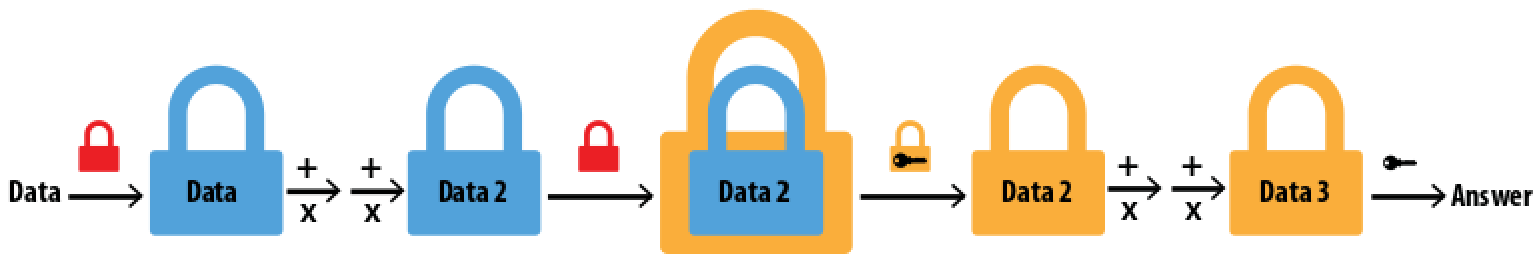
Partially Homomorphic Encryption

Several other encryption schemes are known to be partially homomorphic, including:

- **Goldwasser–Micali Encryption Scheme**
  - Supports addition
- **ElGamal Encryption Scheme**
  - Supports multiplication
- **Paillier Encryption Scheme**
  - Supports addition
- **Boneh–Goh–Nissim Encryption Scheme**
  - Supports unlimited number of additions but only one multiplication (**somewhat** homomorphic cipher).

Fully Homomorphic Encryption

- The first known fully homomorphic cipher was proposed by Craig Gentry in his thesis “*A Fully Homomorphic Encryption Scheme*” [3] in 2009.
- Gentry's construction uses a lattice-based cipher as a starting point.
- Added support for an arbitrary number of additions and multiplications makes the scheme fully homomorphic.
- The scheme is computationally very inefficient.



Source “Securing the cloud with homomorphic encryption”, *The Next Wave*, NSA 2014

Fully Homomorphic Encryption

“*...performing a Google search with encrypted keywords - a perfectly reasonable simple application of this algorithm - would increase the amount of computing time by about a trillion.*”

- Bruce Schneier

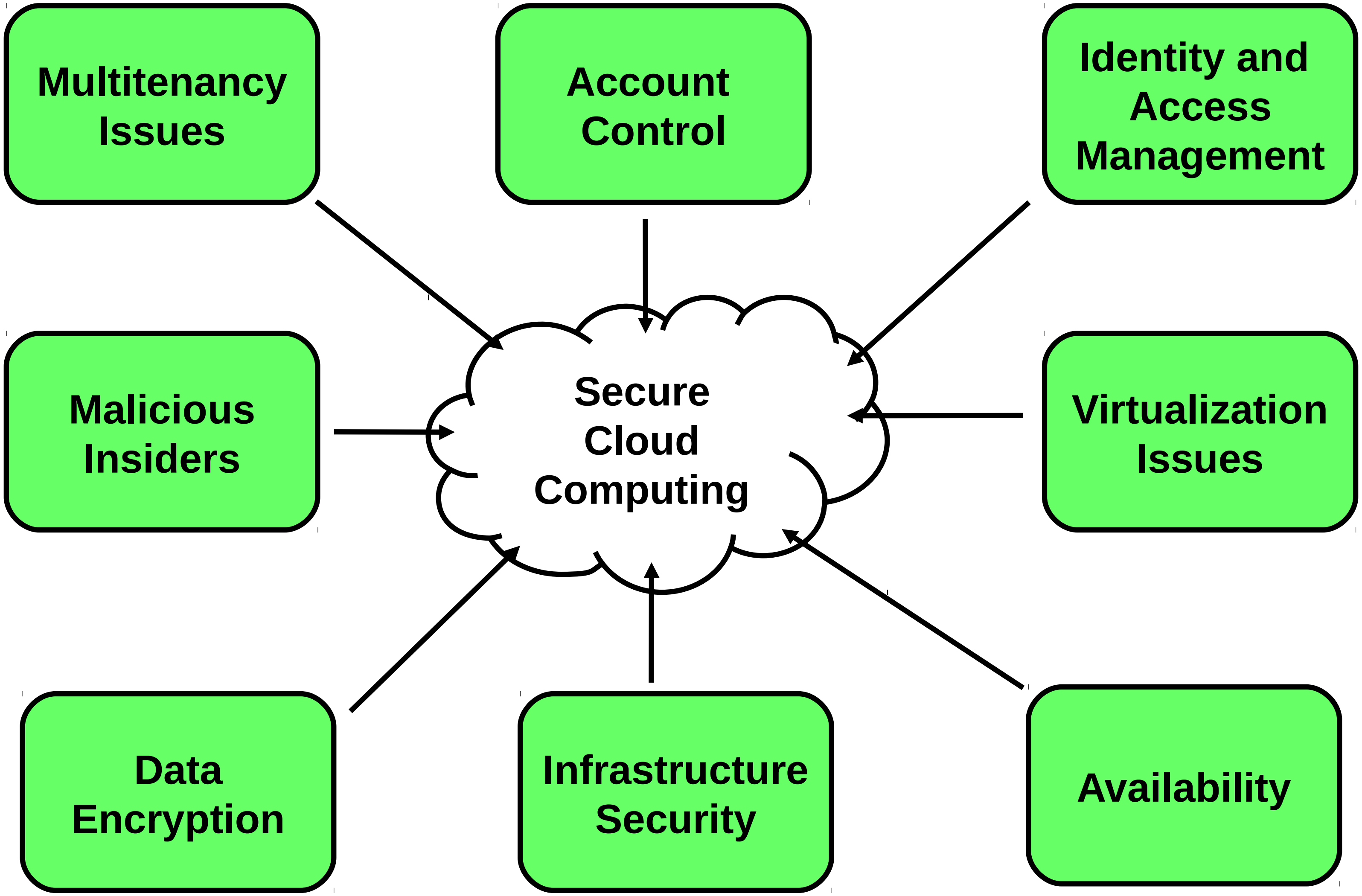
Efficient Fully Homomorphic Encryption

- Several implementations building on Gentry's findings has proved to be more efficient.
- In 2013, Fujitsu launched an efficient scheme for homomorphic encryption based on batch encrypting of data instead of bit-by-bit encryption.
- Several open source implementations such as the HElib library and the FHEW library show good performance.
- Bottom line: Efficiency of homomorphic encryption is increasing to a point where it becomes usable.

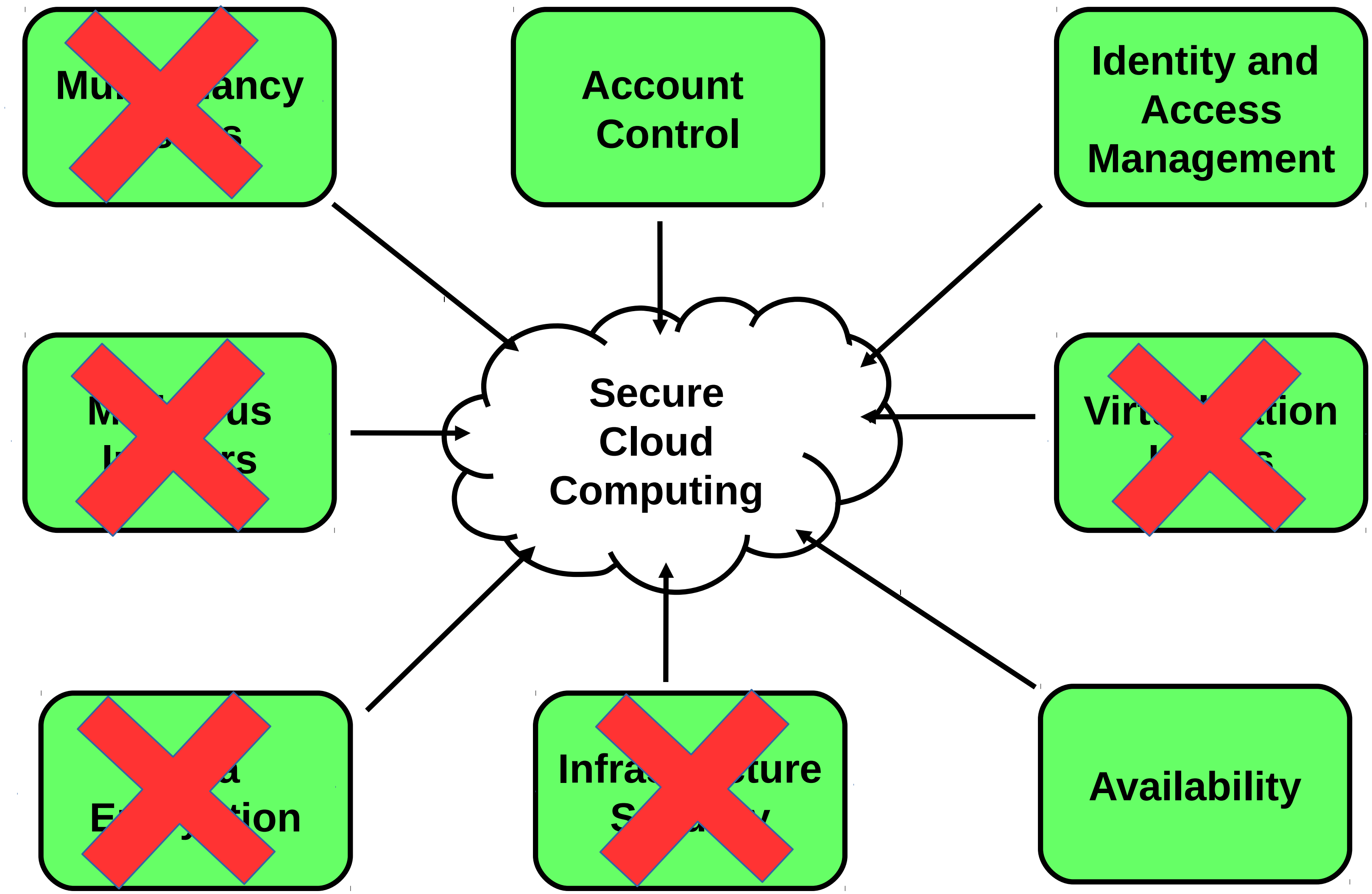
Will this solve all problems?

- If it was possible to apply an efficient, homomorphic encryption scheme, several security issues would be resolved.
- However, many issues would still remain.
An open question is whether the remaining issues are easier to mitigate.

Remaining security issues



Remaining security issues



What will the future hold?

References

1) Zissis, D. and Lekkas, D., 2012. *Addressing cloud computing security issues*. Future Generation Computer Systems, 28(3), pp.583-592.
2) Jansen, W. and Grance, T., 2011. *Guidelines on security and privacy in public cloud computing*. NIST special publication, 800(144), pp.10-11.
3) Gentry, C., 2009. *A fully homomorphic encryption scheme* (Doctoral dissertation, Stanford University).
4) Samarati, P., 2014, May. *Data Security and Privacy in the Cloud*. In ISPEC (pp. 28-41).
5) Yi, X., Paulet, R. and Bertino, E., 2014. *Homomorphic Encryption and Applications* (pp. 1-126). Springer.
6) Brenner, M., Wiebelitz, J., Von Voigt, G. and Smith, M., 2011, May. *Secret program execution in the cloud applying homomorphic encryption*. In Digital Ecosystems and Technologies Conference (DEST), 2011 Proceedings of the 5th IEEE International Conference on (pp. 114-119). IEEE.
7) Yasuda, M., Shimoyama, T., Kogure, J., Yokoyama, K. and Koshiba, T., 2013. *Packed homomorphic encryption based on ideal lattices and its application to biometrics*. In Security Engineering and Intelligence Informatics (pp. 55-74). Springer Berlin Heidelberg.