RISSPA

cloud security alliance SM
CSA Russian Chapter

Шифрование данных в облачных инфраструктурах – обзор технологических подходов

Денис Безкоровайный,
CISA, CISSP, CCSK
Вице-президент RISSPA,
Организатор Cloud Security Alliance Russian Chapter

Нужно ли шифровать данные в облаках?

- Нужно! – говорят сами облачные провайдеры[1]
- Нужно! – говорит Cloud Security Alliance[2]
- Облако провайдера – недоверенная среда
- Помним про US Patriot Act

1 http://awsmedia.s3.amazonaws.com/pdf/AWS_Security_Whitepaper.pdf
2 https://cloudsecurityalliance.org/guidance/csaguide.v3.0.pdf

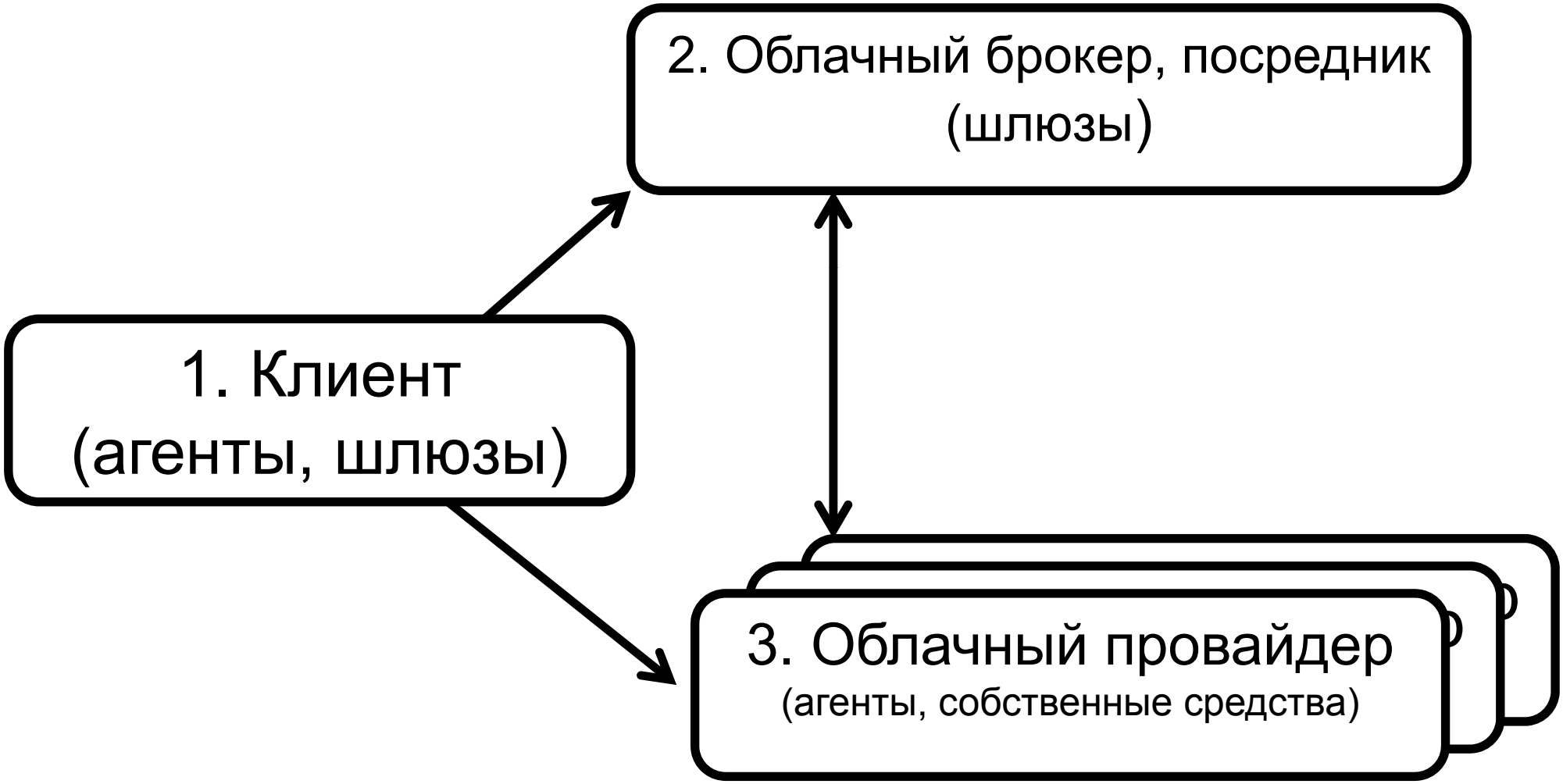
Какие риски нарушения конфиденциальности данных в облаке?

- Доступ к данным со стороны провайдера (злонамеренный инсайдер)
- Публичное разглашение данных (доступ неограниченного круга лиц)
- Вынос/выемка данных или носителей из датацентра провайдера (органы, сотрудники)
- Ошибки изоляции среды (доступ одного клиента облака к данным других клиентов)
- Недостаточное уничтожение данных провайдером при уходе клиента или стирании данных

Варианты подходов к шифрованию в облачных средах

1. Программные агенты на конечных точках, обращающихся к облаку
2. Шлюз/прокси – виртуальный или физический
3. Шифрование непосредственно в самой облачной инфраструктуре

Размещение средств шифрования в облачных средах



Управление ключами шифрования

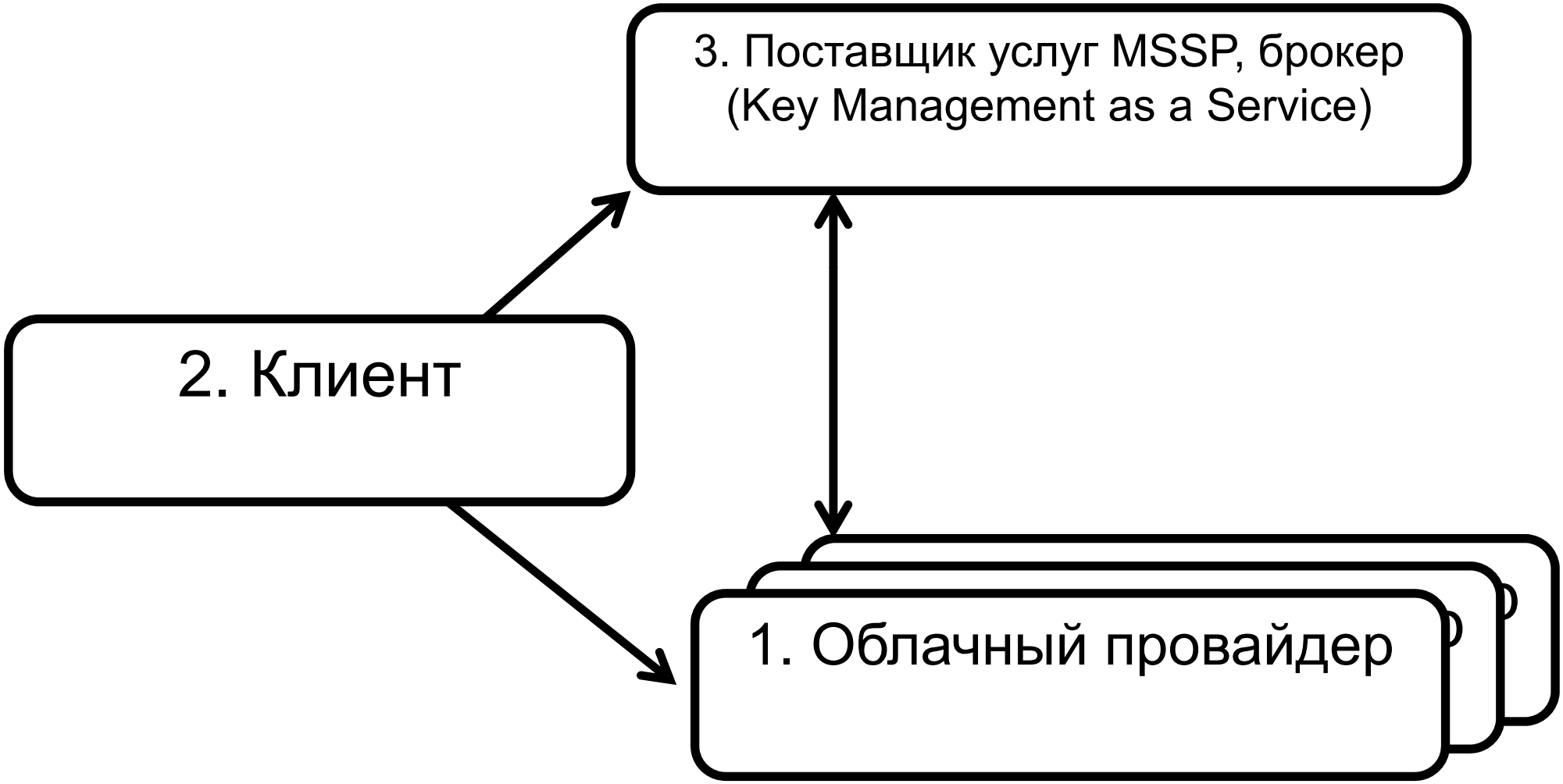
Где хранить ключи?

Как обеспечить защиту ключей?

Как обеспечить доступность ключей?

Сервер управления ключами

Где может находится сервер управления ключами



Сервер управления ключами у провайдера облачной услуги

Минусы

- Нет разделения – и данные и ключи в одних руках
- Риск инсайдера остается
- Риск выемки данных у провайдера остается

Плюсы

- Может быть интегрировано с билингом и сервисами провайдера
- Снижаются риски НСД к данным со стороны других клиентов и риск случайного раскрытия третим лицам

Сервер управления ключами у заказчика облачной услуги

Минусы

- Дорого

Плюсы

- Снижаются все перечисленные риски потери конфиденциальности данных

Сервер управления ключами как услуга

Минусы

- Опять же вопрос доверия к провайдеру услуги
- Дополнительные риски, как и с любым сервисом

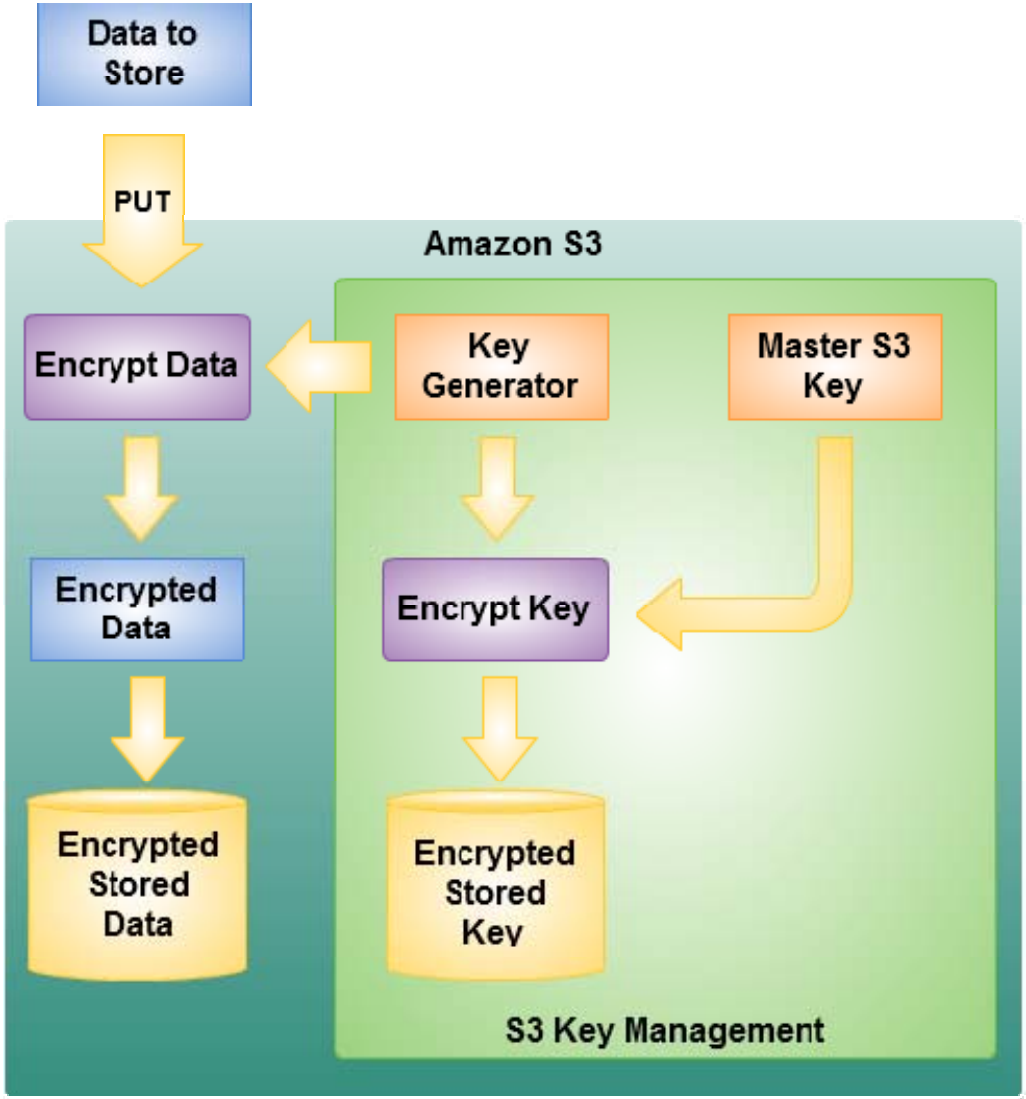
Плюсы

- Разделение данных и ключей
- Снижение всех перечисленных рисков
- Простота
- Относительная дешевизна

Пример реализации шифрования на стороне провайдера - Amazon

Amazon S3 Server Side Encryption for Data at Rest

И средства шифрования и система управления ключами - у провайдера



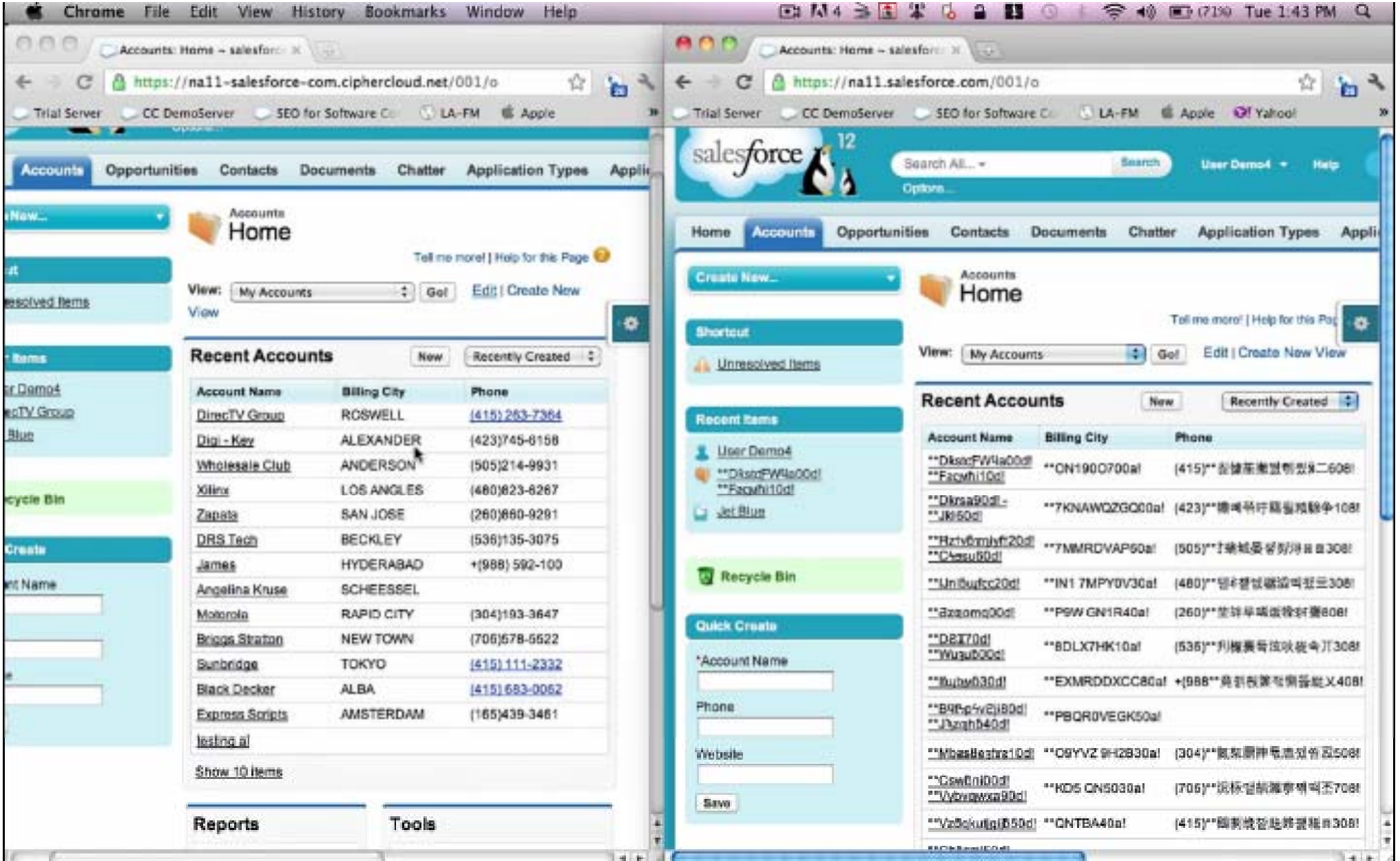
Шифрование для SaaS – реализации

- Шифруются данные полей
- Все или выборочно
  - Агенты на клиентах сервиса
  - Шифрующий/дешифрующий шлюз

Пример реализации шифрования для SaaS - CipherCloud

CipherCloud encrypts sensitive data in real-time, before it's sent to the cloud.

Пример реализации шифрования для SaaS - CipherCloud



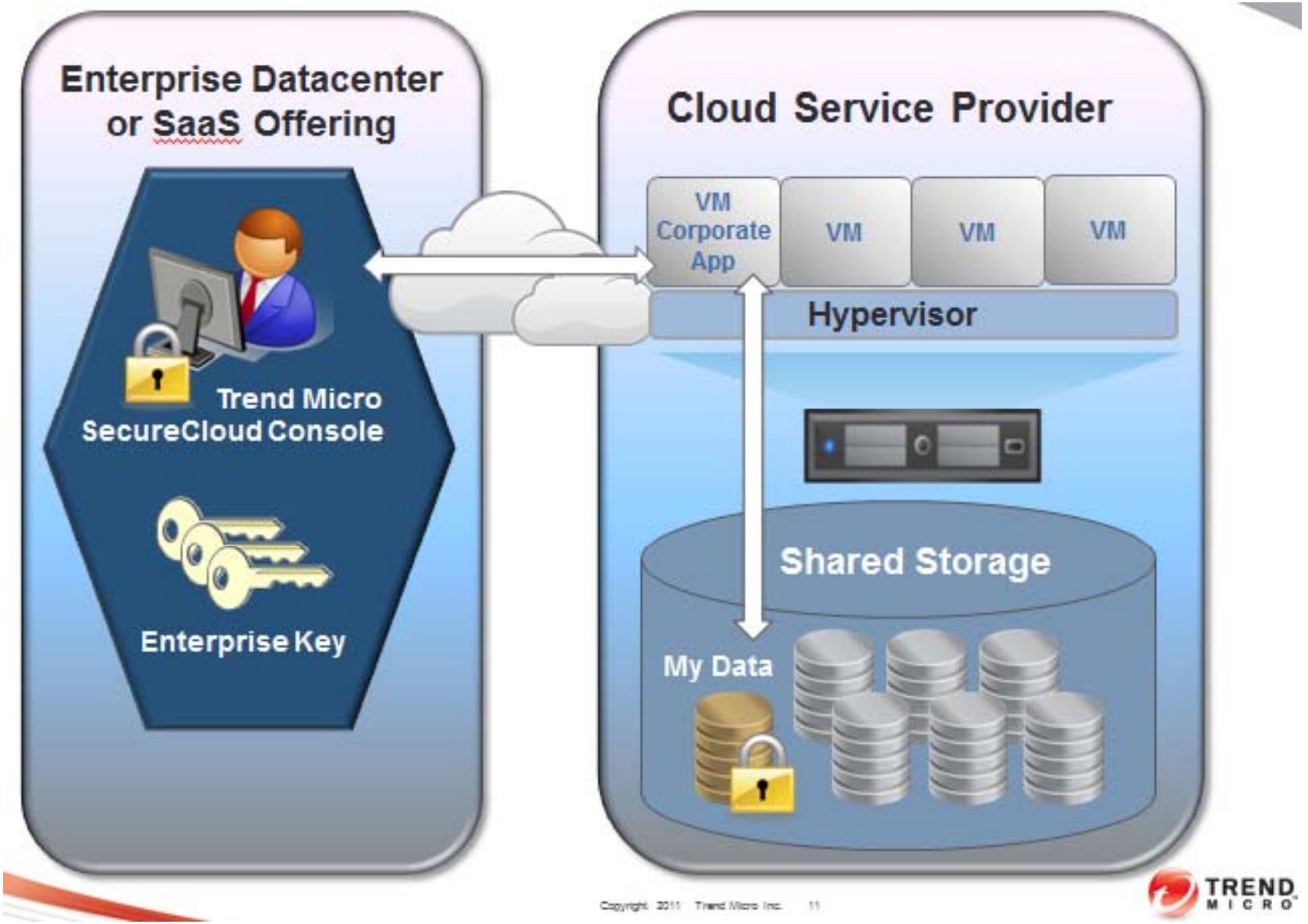
Шифрование для SaaS – сложности

- Обработка данных в приложении (поиск, аналитика)
- Не все поля шифруются
- Нужно либо сохранять форматы данных, либо перестраивать приложения
- Format Preserving Encryption более медленное

Шифрование для IaaS – реализации

- Шифрование инстансов и виртуальных дисков
- Шифрование файлов
- Могут применяться стандартные средства FDE
    - Проблема в аутентификации запроса и получении ключа при загрузке инстанса (нет интерактивного пользователя)

Шифрование для IaaS – пример реализации Trend Micro SecureCloud



Существующие решения – для кого они?

- Для домашних пользователей

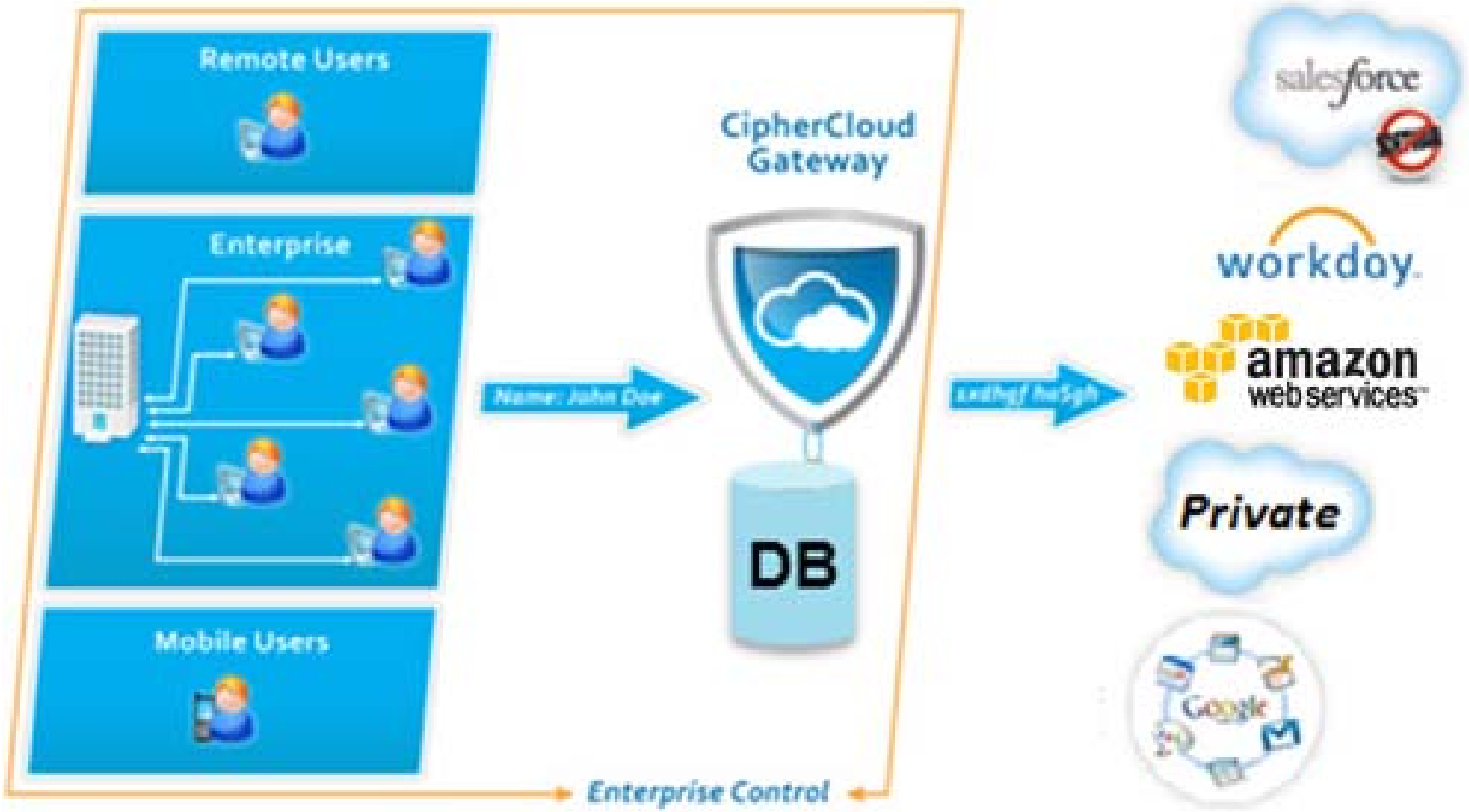
- Для корпоративного сектора (потребители облачных услуг)

- Для облачных сервис провайдеров

Если не шифрование, то что еще?

- Токенизация
- Анонимизация («вырезание» или маскирование конфиденциальных данных)

Пример реализации «облачного токенизатора»



Российские криптоалгоритмы – возможно ли?

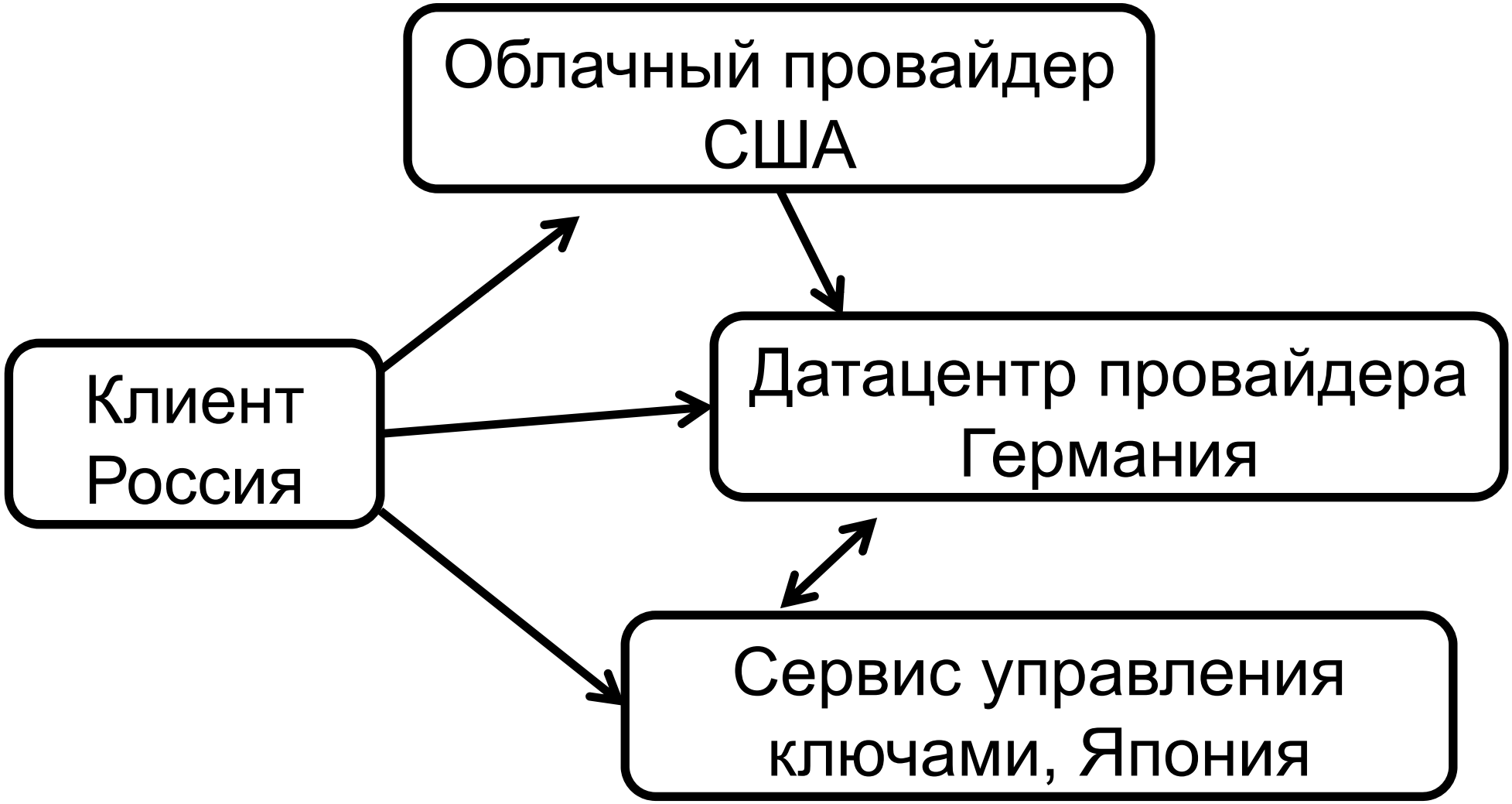
- Готовых решений на рынке нет
- Сделай сам – возможно, например Amazon предоставляет AWS SDK for Java

Crypto Providers

If you want to use a crypto provider other than the default JCE crypto provider, you can specify your own provider with the CryptoConfiguration object:

```
// Bouncy Castle is a third party crypto provider, used as an example.
Provider bcProvider = new BouncyCastleProvider();
CryptoConfiguration cryptoConfig = new CryptoConfiguration().withCryptoProvider(bcProvider);
AmazonS3 encryptionClient = new AmazonS3EncryptionClient(credentials, materials, cryptoConfig);
```

Юридические тонкости



Вопросы?

RISSPA
be professional

bezkod@RISSPA.ru
Денис Безкоровайный

www.RISSPA.ru

LinkedIn