Cloud Data Guard – защита облачных данных!

19.12.2017

burluckij@gmail.com

Сегодня весь современный мир стремительно перетекает в область цифровых пространств, оцифровать можно совершенно всё, крипто-валюты яркий пример. Данные – это “современный доллар”, актив, память, большой объём и разнородность данных позволяют конвертировать их непосредственно в денежную единицу. Каждый отдельный пользователь, а тем более корпорация годами накапливает эти данные (семейные фото, видео, биткоины, потребительская статистика спроса, документы, договора, чеки и т. д.), количество информации неуклонно растёт и каждый клиент (владелец данных) даёт им свою субъективную оценку стоимости. Из чего можно сказать что данные = деньги.

Справедливо сказать -

(1) data is money,

но обратное не выполнимо т.е. -

(2) money не всегда data.

Чтобы money можно было конвертировать в data, требуется посредник, некоторый брокер, сервис, который должен предоставлять, за некоторые деньги (just for money), возможность безопасного доступа к клиентским данным!

Сегодня у нас сложилась уникальная ситуация – ранее все данные хранились на локальных машинах, теперь это можно делать ещё и на смартфонах, планшетах, удалённых компьютерах в сети, докстанциях, внешних сменных носителях информации (жёсткие диски, флешки). Очень разнородно всё, много устройств, разные файловые системы, внутреннее устройство операционных систем, платформ. Решение с совместным доступом к информации – облачные технологии!

Clouds – уникальная возможность иметь доступ к важной тебе информации с любого устройства, 24 на 7 из любой точки планеты. Облака всё больше и больше входят в нашу жизнь, это очень удобно, недорого.

Существующие недостатки в использовании облачных хранилищ

1. Отсутствие доверия со стороны клиентов на честное хранение, сокрытие их информации от третьих лиц. Данные конечно же шифруются на серверной стороне, но провайдер может без ведома клиента предоставить доступ к оригинальным данным для служб безопасности, подвергнуться хакерской атаке.
2. Очень простая возможность доступа к чувствительной информации в момент её нахождения на стороне клиента. Любой злоумышленник, шифровальщик может получить доступ данным.
3. Отсутствие шифрования информации на стороне клиента. Не сервере данные шифруются, у клиента находятся в открытом доступе.
4. Далеко не все провайдеры предоставляют возможность восстановления удалённой, утраченной информации.

Список услуг, которые будут полезны

1. Контроль доступа к данным.
2. Шифрование данных на конечных клиентских машинах.
3. Резервное хранение данных – бэкапы. Очень полезная вещь, можно реализовать несколько способов – зеркалирование облаков, локальные бэкапы и т.д.
4. Безопасное удаление, без возможности восстановления, секретных и важных данных с клиентской машины.
5. Особый интерфейс с повышенной безопасностью для использования и хранения банковской информации в облаке.

Финансовя сторона вопроса

http://www.newsoftwares.net/cloud-secure - простейший локер доступа к папкам облачных хранилищ на локальной машине. 35$ за штуку.

http://www.newsoftwares.net/folderlock - блокировщик доступа к каталогам на диске, включает в себя создание бэкапов данных, шифрование на лету, безопасную очистку данных, затирание свободного места и т.п. Продано более 45 млн. 40$ за штуку.

При цене продукта 30$ за штуку, ожидаются следующие этапы продаж

1000 sales = 30 000 $

10 000 sales = 300 000 $

100 000 sales = 3 000 000 $

1m sales = 30 000 000 $

10m sales = 300m $

В этом сегменте все меряются миллионами продаж, хочу сказать, что 300 тыс. продаж уже делают нас миллионерами долларовыми.

http://www.comss.info/page.php?al=jetapy_razvitija_Avast – история успеха Avast! Как они из очень примитивного технического продукта, выросли в большой коммерчески успешный проект.

https://adguard.com/ru/adguard-windows/overview.html - когда я у них работал, был один и он же флагманский продукт – блокер рекламы под Windows. Сейчас количество платформ выросло, согласно спарку и официальной налоговой информации, за прошлый год, компания задекларировала доход в размере 91 млн руб. Думаю это часть, остальное скрыли.

Открытым остаётся вопрос – лицензия годовая, пожизненная? Тут всё должно быть гибко, нужно предоставлять выбор.

Cloud Data Guard 1.0

Первая версия продукта будет предоставлять следующий функционал:

1. CLAP (CLoud Automatical Protection) - автоматическое определение установленных cloud-клиентов и гибкий контроль доступа к данным (Custom Access Control - CAC).
2. LDAC (Local Data Access Control) – ручное создание защищённых областей (protected areas) на локальном диске с таким же гибким контролем доступа (CAC). Это по сути упрощённая версия CLAP, где пользователю самому необходимо выбирать папку, диск для разграничения доступа.
3. SDER (Secure Data ERase) – безопасная очистка данных. Пользователь может руками из контекстного меню проводника, главного окна нашего приложения удалить список файлов, папок без возможности их последующего восстановления. Данная функция может быть так же интегрирована в CLAP и LDAC, что позволит удалять без возможности восстановления все удалённые файлы с облачного хранилища. Очень часто данные удаляются со смартфона, затем после синхронизации с сервером, они удаляются с десктопов.
4. MFE (Manual Files Encryption) Шифрование файлов – возможность шифрования отдельно выбранных файлов. Сейчас мы не говорим об on-the-fly шифровании, речь идиот о ручном шифровании файлов и папок из главного меню нашего приложения.
5. BSDP (Bank and Sensitive Data Protection) Хранение банковской и особо важной информации о списках кредитных карт и небольших произвольных сообщений.

Все пять пунктов должны быть отражены в меню главного окна графического интерфейса. Это фундамент приложения.

Весь программный комплекс будет состоять из следующих компонентов:

1. Десктопный набор – всё что скрывает в себе установщик.
    a. Системный сервис бизнес логики.
    b. Драйвера режима ядра.
    c. Графический интерфейс.
    d. Библиотеки для контекстного меню проводника Windows.
2. Сервер регистрации клиентов, эквайринг.
3. Выношу в отдельный компонент – база данных всех зарегистрированных пользователей, купивших лицензию.
4. ~~Нотификационный сервис – оповещения о новых версиях.~~ Не в первой версии!

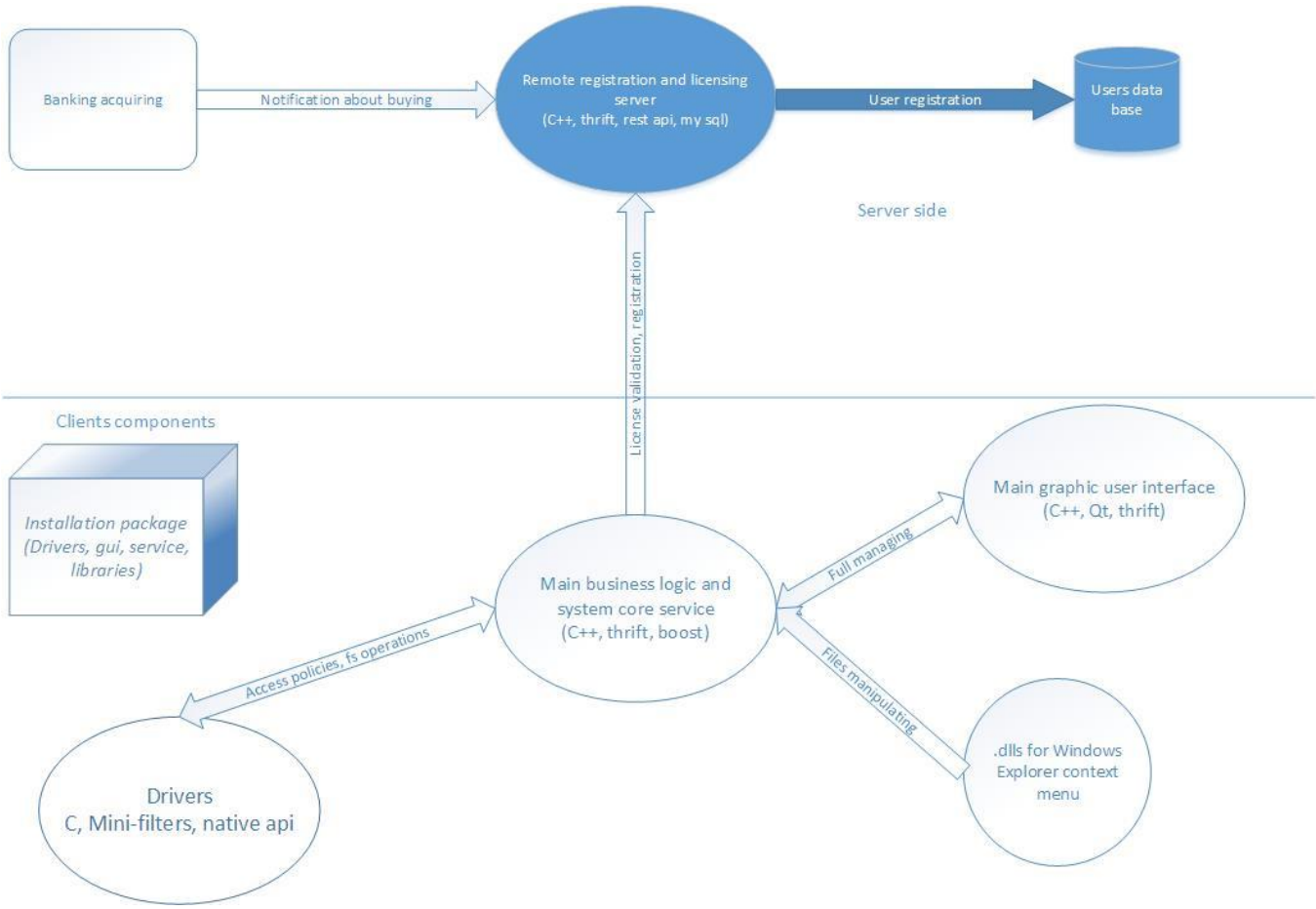

Рисунок 1. Common schema

Клиентская часть

Вся бизнес логика, внутренности, работа с драйверами, проверка лицензии, регистрация – задачи системного сервиса – CDG Service. Все внешнее общение с этим сервисом реализуется посредствам apache::thrift протокола. Он унифицированный, простой, легко расширяемый. В качестве клиентов будут выступать – графический интерфейс, динамические библиотеки для проводника.

Драйвера – управление поведением операционной системы при работе с защищёнными областями, удаление данных.

Графический интерфейс – основной компонент для взаимодействия с пользователем. Отображает состояние системы безопасности, позволяет управлять областями защиты, политиками доступа к облакам и т.д.

Динамические библиотеки – позволяют пользователю удалять данные без возможности восстановления, шифровать содержимое файлов, создавать новые защищённые области данных.

Установщик – содержит все необходимые компоненты для развёртывания и работы системы безопасности у конечного пользователя.

Серверная часть

Сервер регистрации – регистрирует покупки в интернет магазине, создаёт запись о новых пользователях, генерирует лицензионные ключи. С сервером регистрации взаимодействует

банковский клиент и CDG Service запущенный на клиентской машине. CDG Service отправляет информацию о введенном пользовательском ключе, валидирует, позволяет регистрировать новых пользователей, синхронизировать счётчики истечения срока действия лицензии.

База данных пользователей – на самом деле планируется иметь несколько копий одной и той же базы данных – MySql и некоторый самописный примитив на первое время.

Делегирование задач

1. CDG Service – stan
2. Drivers – stan
3. Графический интерфейс – omnio
4. .dll расширения для проводника – omnio
5. Сервер регистрации – stan
6. Серверная база данных – stan, omnio
7. Установщик – stan, omnio

Сайт и эквайринг выполняются в последнюю очередь, после завершения работы над установщиком.

-

Используемые технологии

C++11 минимум - разработка начнётся в Visual Studio 2013 со временем переедим на 2017.

boost – крайне необходим, от него зависят многие сторонние фреймворки, вся работа с xml, json, zip архивами и т. д.

Crypt++ - для работы с криптой и использованием различных известных алгоритмов.

Apache::thrift – всё взаимодействие – как локальное, так и удалённое производится с использованием этого RPC фреймворка.

Qt 5+ - разработка графического интерфейса должна выполнятся на этом с использованием этой кросс-платформенной библиотеки.

Rest API – для обработки запросов от внешних клиентов, вроде банков.

Wix – разработка дистрибутива установщика. Он не простой, но очень популярный и хорошо интегрирован в систему.

Целевые системы, совместимость

Программное обеспечение должно быть совместимо с операционной системой Windows всех версий, начиная с Windows Vista – [Vista, 7, 8, 8.1, 10].

Могут возникнуть сложности с совместимостью для Vista и Win 7 без service pack 1, что может быть вызвано с типами сертификатов для подписывания драйверов. Это нужно обойти, покупкой сертификатов нескольких типов, с расширенной совместимостью.

Этапы, цели, задачи

1. Написать комплекс драйверов с сопутствующими им консольными утилитами для возможности автоматизированного и ручного тестирования.
2. Параллельно драйверам, необходимо вести разработку макета графического интерфейса, для разработки собственного уникального стиля, удобства и внешней пользовательской привлекательности.
3. После драйверов, требуется реализовать каркас сервиса бизнес логики, реализовать взаимодействие с драйверами и предоставить ряд консольных утилит для управления сервисом.
4. После успешного консольного тестирования сервиса бизнес логики, предоставить рабочий, согласованный thrift'овый интерфейс для интеграции с графическим приложением.
5. Использование в графическом интерфейсе предоставленного сервисом программного api.
6. Разработка .dll библиотек для проводника Windows с использованием сервиса бизнес логики.
7. Разработка сервера регистрации, взаимодействие с банком, хранение, предоставление информации о совершенных покупках.
8. Разработка, настройка, интеграция с базой данных в рамках сервера регистрации.
9. Написание установщика.
10. Тестирование установщика, итогового продукта.
11. Покупка сервера на Microsoft Azure с выделенным доменом, статическим IP.
12. Разворачивание серверной инфраструктура на купленном сервере.
13. Регистрация ООО.
14. Покупка цифровых подписей.
15. Регистрация у банка, эквайринг, тестирование проходимости платежей с банка к нашему серверу.
16. Заказ сайта на разработку.
17. Комплексное тестирование.
18. Выравнивание шероховатостей, повтор комплексного тестирования.
19. Продажи!
20. Покупка яхт, самолётов, куртизанки с black jack'ом =))

Далее я продолжаю детализировать комплекс работ, некоторые временные оценки общие буду давать, самые общие. Какие-то конкретные сроки будут после разработки драйверов и макета графического приложения...