Ежегодная международная научно-практическая конференция
«РусКрипто’2019»

Безопасные и расширяемые смарт-контракты в Мастерчейн

Цветков Алексей,
ведущий разработчик, Ассоциация Финтех

Цели и задачи Ассоциации Финтех

Разрабатываем и внедряем новые технологические решения, которые содействуют развитию финансового рынка РФ

Координируем разработку программного обеспечения, стандартов, платформ и протоколов

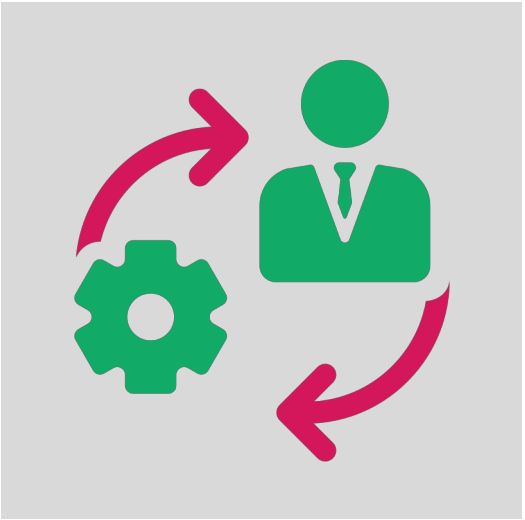
Готовим предложения по созданию и изменению законодательства в области цифровой экономики

ФИНТЕХ АССОЦИАЦИЯ

Как обеспечить безопасность взаимодействия финансовых институтов?

Участники финансового рынка:

- самостоятельно управляют своей инфраструктурой
- строят бизнес-процессы обособленно



Зрелость процессов обеспечения безопасности сильно различается между организациями.

Примеры финансовых процессов в доверенной цифровой среде

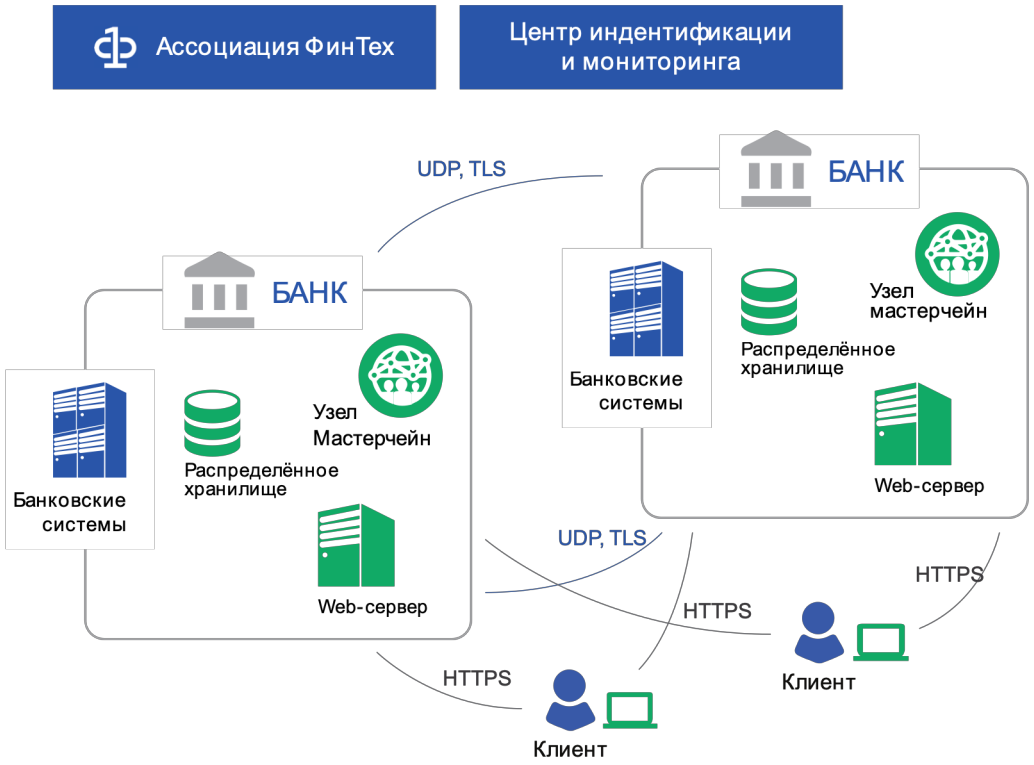
- передача платёжной информации
- регистрация и учет ценных бумаг (электронные закладные)
- торговое финансирование (банковские гарантии, аккредитивы)

Мастерчейн – платформа распределенных реестров для финансовых институтов

На базе ГОСТ-криптографии реализованы:

- механизм исполнения смарт-контрактов
- сервис передачи конфиденциальных сообщений
- средства мониторинга и управления состоянием сети
- создание закрытых сетей с управляемым доступом

Участники сети Мастерчейн



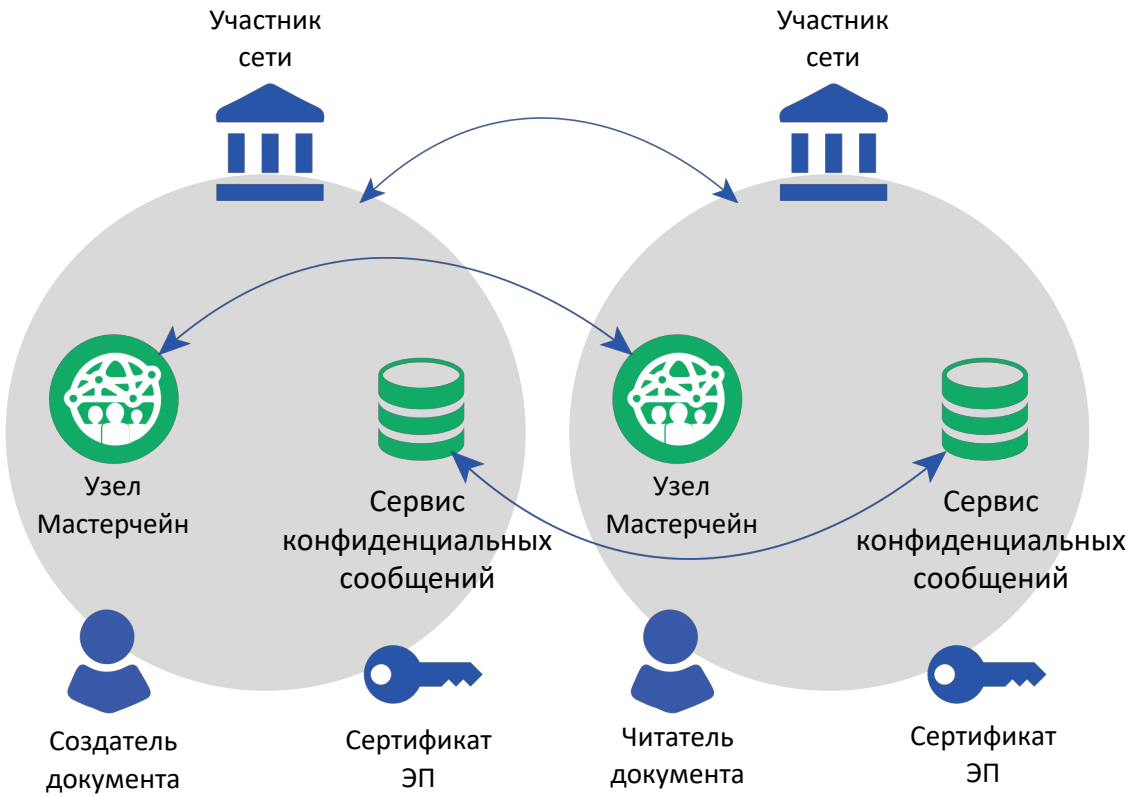
Задачи Ассоциации Финтех:

- разработка и поддержка платформы
- идентификация и лицензирование участников
- мониторинг состояния сети

Участники Ассоциации:

- хранят и ведут реестр
- производят прикладные транзакций

Передача конфиденциальных сообщений



1. Аутентификация в узле сети производится по сертификатам ЭП.
2. В реестре публикуются записи о документах и о правах доступа к ним.
3. Читатель по записи в реестре находит узел и запрашивает документ.
4. Копию документа Читатель сохраняет на своём узле сети.

Смарт-контракты в Мастерчейн

```
// Пример цифрового актива на Мастерчейн
contract Token {

  // Создатель токена (эмитент)
  address public owner;

  // Инициализация токена
  function Token() public {
    owner = msg.sender;
  }

  // Расчётная книга
  mapping(address => uint) public balance;

  // Эмитент может выпустить активы
  function produce(uint amount) public {
    require (msg.sender == owner);
    balance[owner] += amount;
  }

  // Владелец актива может сделать перевод
  function transfer(uint amount, address to) public {
    require (balance[msg.sender] >= amount);
    balance[msg.sender] -= amount;
    balance[to] += amount;
  }
}
```

Примеры уязвимостей смарт-контрактов

```
1  // Пример цифрового актива на Мастерчейн
2  contract Token {
3
4    // Создатель токена (эмитент)
5    address public owner;
6
7    // Инициализация токена
8    function Token() public {
9      owner = msg.sender;
10   }
11
12   // Расчётная книга
13   mapping(address => uint) public balance;
14
15   // Эмитент может выпустить активы
16   function produce(uint amount) public {
17     require (msg.sender == owner);
18     balance[owner] += amount;
19   }
20
21   // Владелец актива может сделать перевод
22   function transfer(uint amount, address to) public {
23     require (balance[msg.sender] >= amount);
24     balance[msg.sender] -= amount;
25     balance[to] += amount;
26   }
27 }
```

необходимо предусмотреть замену ключа владельца

условия для операций будут усложняться

ошибки в операциях с состоянием контракта возможны

- компрометация ключа владельца
- логические ошибки в коде контракта

Контракт состоит из двух частей:

хранимого состояния и логики выполнения.

Обычно механизмы предотвращения атак предназначены для обновления либо состояния, либо логики.

Механизм применения изменений в смарт-контрактах на Мастерчейн

```
1  // Контракт с возможностью отключения
2  contract Suspendable {
3
4      // Флаг отключения
5      bool public disabled;
6
7      // Барьер: отключен ли контракт?
8      modifier enabled() {
9          require(disabled == false);
10         _;
11     }
```

```
12     // Функция проверки прав администратора,
13     // реализуется в дочерних контрактах
14     function canDisable()
15     public view returns(bool);
16
17     // Отключить для изменения
18     function disable() public enabled {
19         require(canDisable());
20         disabled = true;
21     }
22 }
```

Механизм применения изменений в смарт-контрактах на Мастерчейн

```
// Контракт с возможностью обновления
contract Upgradable is Suspendable {

    // Адрес предыдущей версии
    Upgradable public precursor;

    // Адрес следующей версии
    Upgradable public successor;

    // Инициализировать с указанием
    // адреса предыдущей версии
    function Upgradable(Upgradable _precursor)
    public {
        if (_precursor != address(0))
            _precursor.upgradeWith(this);
        precursor = _precursor;
    }
    // Указать адрес следующей версии
    // и отключить
    function upgradeWith(Upgradable _successor)
    public {
        disable();
        successor = _successor;
    }

    // Название смарт-контракта
    function contractName()
    public pure returns(string);

    // Версия смарт-контракта
    function contractVersion()
    public pure returns(string);

}
```

Выводы

Цифровая среда, в которой взаимодействуют участники финансового рынка, нуждается в технических стандартах

Существует технологическая база для создания таких стандартов в виде программного кода

Универсальные механизмы управления средой взаимодействия повышают уровень доверия участников этой среды и общий уровень безопасности

ФИНТЕХ АССОЦИАЦИЯ

Вопросы

Мы приглашаем разработчиков!

Facebook:

facebook.com/fintechassociation

Сайт:

fintechru.org