

제로엑스(0x Protocol)

해당 번역본은 이용자가 디지털자산 관련 정보를 편리하게 확인할 수 있도록
단순 참고용으로 번역된 것이며, 투자 권유를 목적으로 하지 않습니다.
그 내용이 정확하지 않을 수 있으므로 정확한 정보 습득을 위해서는
원문을 참고하시거나 원문 작성 주체 측에 문의하시기 바랍니다.
두나무는 해당 번역본 내용의 진실성 및 정확성을 보장하지 않고,
제공된 정보에 의한 투자 결과에 대하여 어떠한 책임도 지지 않으며,
투자 손실은 투자자에게 귀속됩니다.

0x: 이더리움 블록체인에서 탈중앙화 거래 를 위한 오픈 프로토콜

Will Warren, Amir Bandeali
0xProject.com

2017년 2월 21일

초록

우리는 이더리움 블록체인에서 ERC20 토큰의 낮은 마찰 피어-투-피어 교환을 가능하게 하는 프로토콜을 설명한다. 이 프로토콜은 교환 기능을 포함하는 탈중앙화 애플리케이션(dApp) 간의 상호운용성을 촉진하는 오픈 표준이자 공통 빌딩 블록으로 사용되도록 설계되었다. 거래는 공개적으로 접근 가능하고, 자유롭게 사용 가능하며, 어떤 dApp도 연결할 수 있는 이더리움 스마트 컨트랙트 시스템에 의해 실행된다. 프로토콜 위에 구축된 dApp은 공용 유동성 풀에 접근하거나 자체 유동성 풀을 생성하고 결과 거래량에 대해 수수료를 부과할 수 있다. 이 프로토콜은 특정 의견을 강요하지 않는다: 사용자에게 비용을 부과하거나 임의로 한 사용자 그룹에서 가치를 추출하여 다른 그룹을 이익하게 하지 않는다. 탈중앙화 거버넌스는 dApp이나 최종 사용자를 방해하지 않고 기본 프로토콜에 업데이트를 지속적이고 안전하게 통합하는 데 사용된다.

목차

1. 서론
2. 관련 연구

- 3. 상세
 - 3.1 메시지 포맷
 - 3.1.1 포인트-투-포인트 주문
 - 3.1.2 브로드캐스트 주문
 - 3.2 스마트 컨트랙트
 - 3.2.1 서명 인증
 - 3.2.2 체결 및 부분 체결
 - 3.2.3 만료 시간
 - 3.2.4 주문 취소
- 4. 프로토콜 토큰
 - 4.1 탈중앙화 거버넌스
 - 4.1.1 지속적 통합
 - 4.1.2 토큰 레지스트리
- 5. 요약
- 6. 감사의 글
- 7. 부록
 - 7.1 ERC20 토큰
 - 7.2 컨트랙트 ABI
 - 7.3 이더리움 네임 서비스

1. 서론

블록체인은 누구나 신뢰할 수 있는 제3자 없이도 개방된 금융 네트워크 전반에서 자산을 소유하고 전송할 수 있도록 함으로써 혁신을 가져왔다. 이제 수백 [1] 개의 블록체인 기반 자산이 존재하며 매달 더 많은 자산이 추가되고 있어, 이러한 자산을 교환해야 하는 필요성이 가속적으로 증가하고 있다. 스마트 컨트랙트의 출현으로, 두 명 이상의 당사자가 신뢰할 수 있는 제3자 없이 블록체인 자산을 교환하는 것이 가능해졌다.

탈중앙화 거래소는 중앙화 거래소 생태계에서 중요한 진보이다. 그 이유는 몇 가지가 있는데: 탈중앙화 거래소는 더 이상 중앙 당사자가 존재하지 않으므로 사용자에게 더 강력한 보안 보장을 제공할 수 있다. 중앙 당사자는 해킹을 당하거나 고객 자금을 탈취하거나 정부 규제를 받을 수 있다. Mt. Gox, Shapeshift, Bitfinex [2, 3]의 해킹 사건은 이러한 유형의 시스템적 위험이 실제적임을 보여주었다. 탈중앙화 거래소는 사용자가 신뢰할 수 있는 방식으로 — 즉 중개인 없이 — 거래할 수 있도록 하고, 보안 부담을 단일 수탁자에게서 개별 사용자에게 분산시킴으로써 이러한 위험을 제거할 것이다.

이더리움 블록체인의 제네시스 블록이 생성된 이후 지난 2년 동안, 수많은 탈중앙화 애플리케이션(dApp)이 피어-투-피어 거래를 위한 이더리움 스마트 컨트랙트를 만들어왔다. 빠른 반복과 모범 사례 부족으로 인해 블록체인은 독점적이고 애플리케이션 특화된 구현체들로 흩어졌다. 그 결과, 최종 사용자는 품질과 보안이 제각각인 수많은 스마트 컨트랙트에 노출되며, 고유한 설정 절차와 학습 곡선을 요구받는데, 이 모든 것이 동일한 기능을 구현하고 있다. 이러한 접근 방식은 사용자들을 특정 dApp에 따라 분절시켜 네트워크에 불필요한 비용을 부과하고, 유동성과 관련된 귀중한 네트워크 효과를 파괴한다.

0x는 이더리움 블록체인에서의 탈중앙화 거래를 위한 오픈 프로토콜이다. 이는 점점 더 정교해지는 dApp [4]을 구동하기 위해 다른 프로토콜과 결합될 수 있는 기본 빌딩 블록 역할을 하도록 설계되었다. 0x는 그림 1에 나타난 것처럼 다양한 dApp을 위한 공유 인프라로 기능할 수 있는 공개 접근 가능한 스마트 컨트랙트 시스템을 사용한다. 장기적으로 개방형 기술 표준은 폐쇄형 표준보다 우위를 점하는 경향이 있으며, 매달 더 많은 자산이 블록체인에서 토큰화됨에 따라, 우리는 이러한 다양한 토큰 사용을 요구하는 더 많은 dApp을 보게 될 것이다. 따라서 교환을 위한 개방형 표준은 이 개방 경제를 지원하는 데 필수적이다.

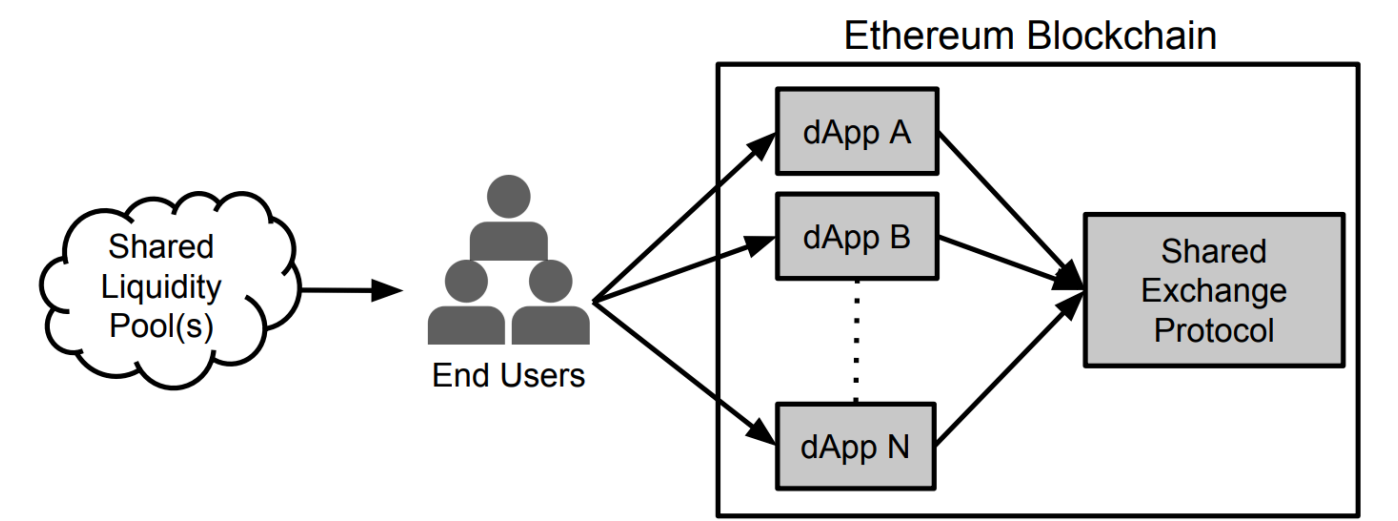


그림 1: 오픈 프로토콜은 애플리케이션에 구매받지 않아야 한다. 프로토콜 계층을 애플리케이션 계층과 분리 하면 dApp 개발자와 최종 사용자 모두에게 상호 이익을 제공한다.

2. 관련 연구

이더리움 스마트 컨트랙트로 구현된 탈중앙화 거래소는 시장 메이커에게 높은 마찰 비용을 부과하는 설계 비 효율성 때문에 상당한 거래량을 생성하지 못했다. 특히 이러한 구현체들은 주문서를¹ 블록체인 위에 두어 [5–8], 시장 메이커가 주문을 게시, 수정 또는 취소할 때마다 가스를 소비하도록 요구한다. 단일 거래의 비용은 작지만, 변화하는 시장 상황에 대응하여 주문을 자주 수정하는 것은 금지될 정도로 비용이 많이 든다. 시장 메이커에게 높은 비용을 부과하는 것 외에도, 온체인 주문서를 유지하는 것은 반드시 가치 이전을 수반하지 않으면서도 네트워크 대역폭을 소모하고 블록체인을 비대하게 만드는 거래를 발생시킨다.

(¹ 주문서는 특정 금융 상품에 대한 매수자와 매도자의 관심을 공개적으로 기록하는 데 사용된다. 각 항목에는 이해 당사자, 수량, 그리고 매수자 또는 매도자가 해당 증권에 대해 입찰/호가하는 가격이 포함된다.)

온체인 주문서의 대안으로 자동화된 마켓 메이커(AMM) 스마트 컨트랙트가 제안되었다 [9, 10]. AMM 스마트 컨트랙트는 주문서를 가격 조정 모델로 대체하며, 이 모델에서 자산의 현물 가격은 시장의 힘에 결정적으로 반응하고 시장 참여자는 서로가 아닌 AMM과 거래한다. AMM의 장점에는 가용성(항상 거래 상대방 역할을 할 수 있지만, 제공하는 현물 가격은 전통적인 거래소에서 얻을 수 있는 것보다 나쁠 수 있음)과 시장가 주문을 실행해야 하는 외부 스마트 컨트랙트와의 통합 용이성이 포함된다. 가격 조정 모델의 결정론적 특성은 이를 시장 유동성에 둔감하게 만들어, 거래가 유동성이 높은 시장과 낮은 시장 모두에서 동일한 폭으로

가격을 움직이게 한다 [11]. 다시 말해, AMM은 공급 곡선에 인위적인 제약을 부과한다. 가격 조정 모델이 지나치게 민감하면 작은 거래조차 현물 가격의 큰 변동을 초래한다. 반대로 가격 조정 모델이 충분히 민감하지 않으면, AMM의 자금은 차익거래자들에 의해 빠르게 고갈될 것이다.

상태 채널은 거래를 블록체인 밖으로 이동시킴으로써 [13, 14], 이더리움 블록체인을 확장하고 다양한 애플리케이션(거래소 포함 [12])의 비용을 줄이는 수단으로 제안되었다. 상태 채널의 참가자들은 암호학적으로 서명된 메시지를 주고받으며, 채널이 닫힐 때까지 이를 정식 체인에 게시하지 않고 중간 상태 변화를 축적한다. 상태 채널은 수많은 중간 상태 변화를 온체인 거래 한 건으로 정산하기 전에 오프체인에서 누적할 수 있는 “바 탭(bar tab)” 애플리케이션에 이상적이다(예: 데이 트레이딩, 포커, 던제 게임). 채널 참가자 중 한 명이 채널을 떠나거나 속임수를 쓰려 하면, 다른 참가자가 위반자로부터 받은 가장 최근 메시지를 게시할 수 있는 도전 기간(challenge period)이 존재한다. 따라서 채널 참가자들은 항상 온라인 상태를 유지해야 부정직한 상대방을 제지할 수 있으며, 이로 인해 DDOS 공격에 취약해진다. 상태 채널은 특정 사용 사례에서 온체인 거래 수를 극적으로 줄이지만, 상태 채널을 열고 안전하게 닫기 위해 필요한 수많은 온체인 거래와 보증금 때문에 일회성 거래에는 비효율적이다.

우리가 “오프체인 주문 릴레이와 온체인 결제”라고 부르는 하이브리드 구현은 상태 채널의 효율성과 온체인 주문서의 거의 즉각적인 결제를 결합한다. 이 접근 방식에서는 암호학적으로 서명된 주문이 블록체인 밖으로 브로드캐스트되고, 관심 있는 상대방은 이러한 주문 중 하나 이상을 스마트 컨트랙트에 삽입하여 블록체인 상에서 신뢰할 수 없는 방식으로 직접 거래를 실행할 수 있다 [15, 16]. 시장 메이커는 오프체인에서 의사를 신호할 수 있고 실제 가치 이전이 발생할 때만 거래가 이루어지므로 마찰 비용이 최소화된다. 우리는 이 접근 방식을 확장하여 누구나 거래소 역할을 수행할 수 있도록 하고 프로토콜을 애플리케이션에 구애받지 않도록 만든다.

3. 상세

그림 2는 오프체인 주문 릴레이와 온체인 결제에 사용되는 일반적인 절차 순서를 보여준다. 지금은 나중에 중요해질 몇 가지 메커니즘은 무시한다.

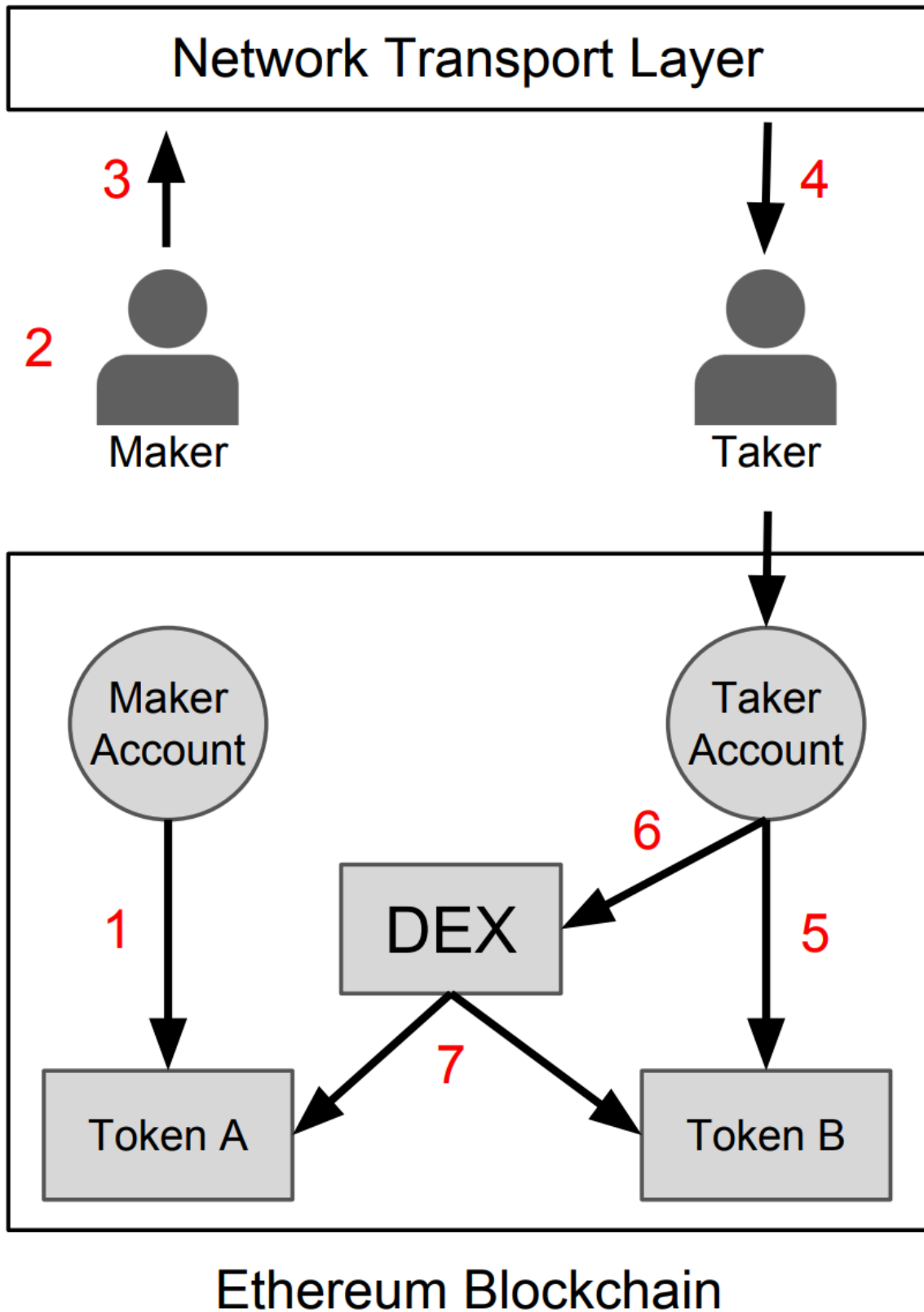


그림 2: 오프체인 주문 릴레이, 온체인 결제 다이어그램. 회색 사각형과 원은 각각 이더리움 스마트 컨트랙트와 계정을 나타낸다. 이더리움 스마트 컨트랙트를 가리키는 화살표는 함수 호출을 의미하며, 화살표는 호출자에서 피호출자로 향한다. 스마트 컨트랙트는 다른 스마트 컨트랙트 내의 함수를 호출할 수 있다. 이더리움 블록체인의 외부의 화살표는 정보 흐름을 나타낸다.

1. 메이커는 탈중앙화 거래소(DEX) 컨트랙트가 토큰 A² 잔액에 접근할 수 있도록 승인한다.

2. 메이커는 토큰 A를 토큰 B로 교환하기 위한 주문을 생성하고, 원하는 환율과 만료 시간(만료 이후에는 주문을 체결할 수 없음)을 지정한 뒤, 개인 키로 주문에 서명한다.
3. 메이커는 임의의 통신 매체를 통해 주문을 브로드캐스트한다.
4. 테이커는 주문을 수신하고 이를 체결하기로 결정한다.
5. 테이커는 DEX 컨트랙트가 자신의 토큰 B 잔액에 접근할 수 있도록 승인한다.
6. 테이커는 메이커가 서명한 주문을 DEX 컨트랙트에 제출한다.
7. DEX 컨트랙트는 메이커의 서명을 인증하고, 주문이 만료되지 않았음을 검증하며, 주문이 이미 체결되지 않았음을 확인한 후, 지정된 환율에 따라 두 당사자 간에 토큰을 전송한다.

(² 부록의 ERC20 토큰을 참조하라. 한 번의 승인으로 이후 무제한 거래를 실행하는 것이 가능하다. 또는 각 개별 거래 전에 해당 거래 금액에 한정하여 승인을 제공할 수도 있다.)

3.1 메시지 포맷

각 주문은 주문 파라미터와 관련 서명을 포함하는 데이터 패킷이다. 주문 파라미터는 연결(concatenate)되어 Keccak SHA3 함수로 32바이트 해시로 변환된다. 주문 생성자는 개인 키로 주문 해시에 서명하여 ECDSA 서명을 생성한다.

3.1.1 포인트-투-포인트 주문

포인트-투-포인트 주문은 두 당사자가 선호하는 거의 모든 메시지 전달 매체를 사용하여 서로 직접 토큰을 교환할 수 있도록 한다. 주문을 구성하는 데이터 패킷은 수백 바이트의 헥스 문자열이며 이메일, 페이스북 메시지, Whisper 또는 이와 유사한 서비스를 통해 전송될 수 있다. 주문은 지정된 *테이커* 주소에 의해서만 체결될 수 있으며, 이는 도청자나 외부 당사자에게는 무용지물이 된다.

표 1: 포인트-투-포인트 주문의 메시지 포맷.

이름	데이터 타입	설명
version	address	거래소 스마트 컨트랙트의 주소. 프로토콜이 업데이트될 때마다 이 주소는 변경된다.
maker	address	주문을 생성한 주소.
taker	address	주문을 체결할 수 있는 허가된 주소.
tokenA	address	ERC20 토큰 컨트랙트의 주소.
tokenB	address	ERC20 토큰 컨트랙트의 주소.
valueA	uint256	메이커가 제공하는 tokenA의 총 단위.
valueB	uint256	메이커가 요청하는 tokenB의 총 단위.
expiration	uint256	주문이 만료되는 시간(유닉스 에포크 이후 초 단위).
v	uint8	위의 인자들에 대한 ECDSA 서명.
r	bytes32	
s	bytes32	

3.1.2 브로드캐스트 주문

유동적인 시장이 형성되기 위해서는 매수자와 매도자가 주문을 게시하고 이후 주문서로 집계될 수 있는 공개 장소, 즉 거래소가 필요하다. 거래소를 구축하고 운영하는 것은 비용이 많이 들며, 지금까지 설명한 프로토콜은 누군가가 그러한 비용을 감수하도록 유인하지 않는다. 브로드캐스트 주문은 누구나 거래소 역할을 수행하고, 주문서(공개 또는 비공개)를 유지하며, 결과적으로 발생하는 유동성에 대해 거래 수수료를 부과할 수 있도록 함으로써 이 문제를 해결한다. 우리는 주문서를 호스팅하고 유지하는 개체를 거래소가 아니라 릴레이어(Relayer)라고 부른다. 거래소는 독점 인프라를 구축하고 운영하며, 거래를 실행하고, 사용자 자금을 관리해야 하지만, 릴레이어는 일반 메시지로 구성된 주문서를 호스팅하고 전파하여 시장 참여자 간 신호를 촉진할 뿐이다. 릴레이어가 시장 참여자를 대신해 거래를 실행하는 일은 없는데, 이는 시장 참여자가 릴레이어를 신뢰해야 하기 때문이다. 대신, 테이커는 자신의 거래를 직접 실행한다.

브로드캐스트 주문의 메시지 포맷은 공개 거래를 용이하게 하고 릴레이어에게 인센티브를 제공하기 위해 포인트-투-포인트 메시지 포맷에 두 가지 변경 사항을 포함한다. 첫째, 브로드캐스트 주문은 *테이커* 주소를 지정하지 않으므로, 이를 수신한 누구든지 주문을 체결할 수 있다. 둘째, 브로드캐스트 주문에는 *feeA*, *feeB*, *feeRecipient* 파라미터가 포함되어 있으며, 이는 거래 수수료 값과 릴레이어가 거래 수수료를 수취하는 데

사용하는 주소를 지정한다. 거래소 스마트 컨트랙트는 주문이 체결되면 이 수수료를 *feeRecipient*에게 전송한다. 그림 3은 메이커와 릴레이어가 신뢰 없는 방식으로 거래 수수료를 협상하는 데 사용하는 절차를 보여준다.

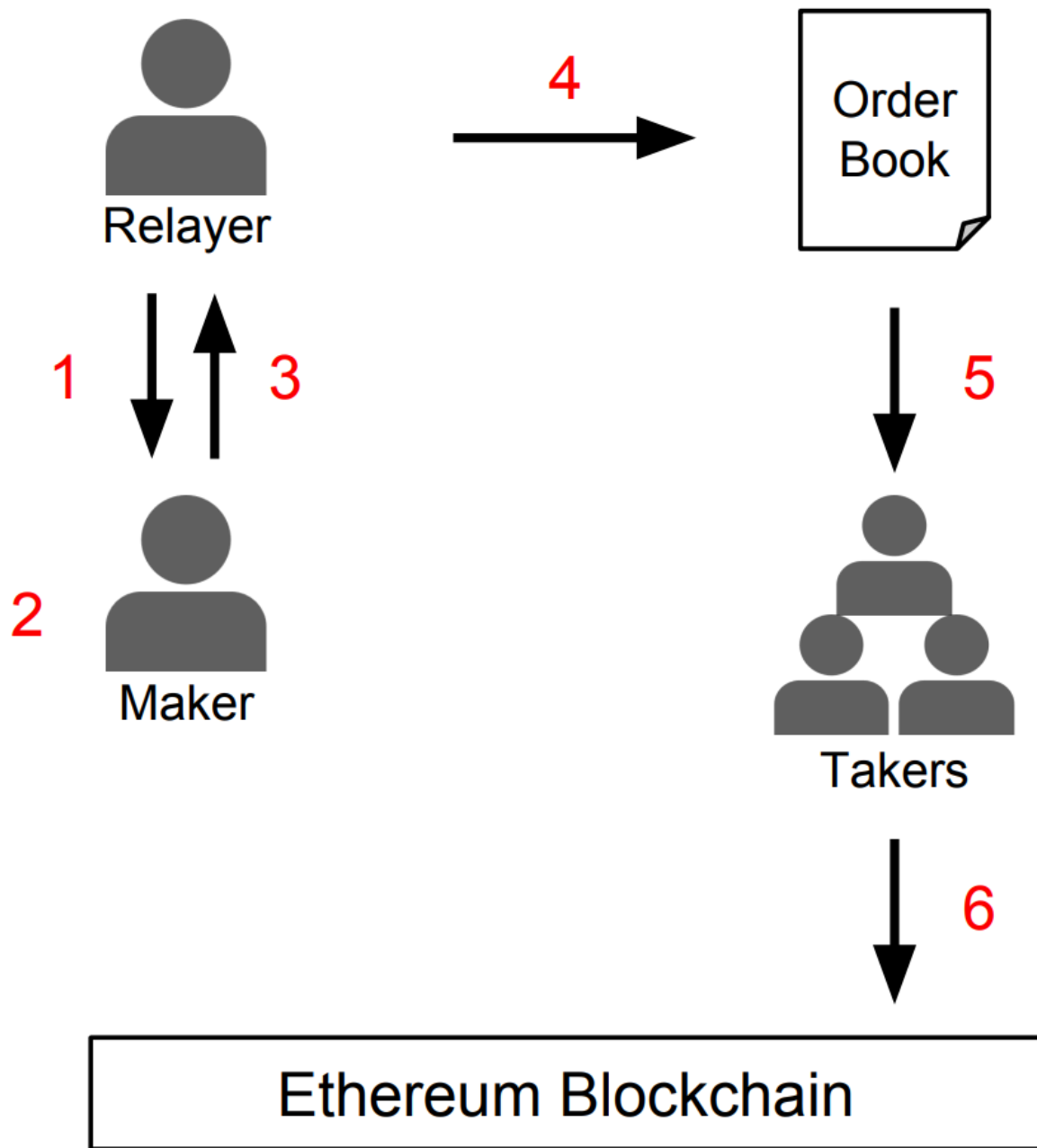


그림 3: 릴레이어는 거래 수수료와 맞바꾸어 오프체인 주문서를 호스팅하고 유지한다. 이 다이어그램은 오프체인 주문 릴레이와 메이커 및 릴레이어가 신뢰 없는 방식으로 거래 수수료를 협상하는 데 사용하는 절차를 보여준다. 거래 수수료는 거래 정산 시 메이커와/또는 테이커로부터 릴레이어로 이동하며, 이는 그림 2에 나타난 온체인 정산 프로세스를 확장한 것이다.

1. 릴레이어는 거래 수수료 체계와 거래 수수료를 수취하는 데 사용하는 주소를 명시한다.
2. 메이커는 릴레이어의 수수료 체계를 충족하도록 *feeA*와 *feeB* 값을 설정하고, *feeRecipient*를 릴레이어가 원하는 수취 주소로 지정한 후 개인 키로 주문에 서명한다.
3. 메이커는 서명된 주문을 릴레이어에게 전송한다.

4. 릴레이어는 주문을 수신하고 주문이 유효하며 요구된 수수료를 제공하는지 확인한다. 주문이 유효하지 않거나 릴레이어의 요구 사항을 충족하지 않으면 주문은 거부된다. 주문이 만족스러우면 릴레이어는 주문을 자신의 주문서에 게시한다.
5. 테이커는 메이커의 주문이 포함된 업데이트된 주문서를 수신한다.
6. 테이커는 메이커의 주문을 이더리움 블록체인 상의 거래소 컨트랙트에 제출하여 주문을 체결한다.

표 2: 브로드캐스트 주문의 메시지 포맷.

이름	데이터 타입	설명
version	address	거래소 스마트 컨트랙트의 주소.
maker	address	주문을 생성한 주소.
tokenA	address	ERC20 토큰 컨트랙트의 주소.
tokenB	address	ERC20 토큰 컨트랙트의 주소.
valueA	uint256	메이커가 제공하는 tokenA의 총 단위.
valueB	uint256	메이커가 요청하는 tokenB의 총 단위.
expiration	uint256	주문이 만료되는 시간(유닉스 에포크 이후 초 단위).
feeRecipient	address	릴레이어의 주소. 거래 수수료를 수취한다.
feeA	uint256	메이커가 <i>feeRecipient</i> 에게 지불하는 프로토콜 토큰의 총 단위.
feeB	uint256	테이커가 <i>feeRecipient</i> 에게 지불하는 프로토콜 토큰의 총 단위.
v	uint8	위 인자들에 대한 ECDSA 서명.
r	bytes32	
s	bytes32	

메이커가 거래 수수료를 지정하는 것이 다소 이상하게 보일 수 있지만, 궁극적으로 어떤 주문이 게시될지를 통제하는 것은 릴레이어라는 점을 유념해야 한다. 따라서 메이커가 자신의 주문을 특정 주문서에 게시되도록 하려면, 해당 주문서와 관련된 릴레이어를 만족시키는 값으로 *feeA*, *feeB*, *feeRecipient*를 설정해야 한다. 수수료는 오프체인에서 협상되므로, 릴레이어는 자신의 재량에 따라 동적으로 수수료 체계를 변경할 수 있다 (이미 서명된 기존 주문이 아닌, 아직 서명되지 않은 신규 주문에 대해서만 가능). 릴레이어는 온체인 또는 오프체인에서 사용할 수 있는 정보를 활용해 수수료를 설정 및 조정할 수 있으며, 이를 통해 유연한 수수료 체계

(정액제, 비율 기반, 거래량 기반, 계층형, 구독 모델 등)를 적용할 수 있다. 그러나 일단 릴레이어가 주문을 자신의 주문서에 수락하면, 해당 주문의 수수료 값은 변경될 수 없다.

전통적인 거래소 서비스는 매칭 엔진을 사용하여 사용자 대신 시장가 주문을 체결하며, 사용자는 거래소가 최상의 가격을 제공할 것이라는 점을 신뢰해야 한다. 일반적으로 사용자는 이러한 규제된 기관이 속임수를 시도하거나 매칭 엔진이 오작동할 경우 책임을 질 것이라 안심할 수 있다. 그러나 0x 프로토콜이 신뢰 없는 (trustless) 상태를 유지하려면, 릴레이어에게 메이커와 테이커를 대신하여 거래를 실행할 권한을 부여할 수 없다. 대신 릴레이어는 테이커에게 최상의 이용 가능한 가격을 추천할 수 있으며, 테이커는 이후 독립적으로 서명하고 거래를 블록체인에 전송할지를 결정해야 한다. 이는 0x 프로토콜이 진정한 의미의 시장가 주문을 지원할 수 없음을 의미하지만, 잘 설계된 웹 애플리케이션은 이러한 사용자 경험을 유사하게 구현할 수 있다.

feeRecipient 주소가 임의의 스마트 컨트랙트를 가리킬 수 있다는 점을 인식하는 것이 중요하다. 이는 복잡한 릴레이어 인센티브 구조가 0x 프로토콜에 “플러그인”될 수 있음을 의미한다. 예를 들어, *feeRecipient* 컨트랙트는 여러 릴레이어 간에 거래 수수료를 분배하거나, 검열 저항적인 p2p 네트워크 내에서 주문서를 전파하는 데 각 노드가 기여한 수준에 따라 거래 수수료를 노드 집합에 분산시키도록 설계될 수 있다³.

(³ 완전히 분산된 주문서를 지원하는 저지연 릴레이 프로토콜의 개발이 이 프로젝트의 다음 단계로 고려되고 있다.)

3.2 스마트 컨트랙트

거래 프로토콜은 공개적으로 접근 가능하고 자유롭게 사용할 수 있는 이더리움 스마트 컨트랙트 내에 구현되어 있다(사용자에게 표준 가스 비용 이외의 추가 비용은 부과되지 않는다). 이 컨트랙트는 Solidity 프로그래밍 언어로 작성되었으며, 상대적으로 단순한 두 개의 함수: *fill*과 *cancel*을 포함한다. 전체 컨트랙트는 약 100줄의 코드로 구성되며, 주문을 체결하는 데 약 90k 가스가 소요된다.

3.2.1 서명 인증

거래소 스마트 컨트랙트는 *ecrecover* 함수를 사용하여 주문 생성자(메이커)의 서명을 인증할 수 있다. 이 함수는 해시와 그 해시에 대한 서명을 인자로 받아 서명을 생성한 공개 키를 반환한다. *ecrecover*가 반환한 공개 키가 *maker* 주소와 동일하다면, 해당 서명은 진본이다.

```
address publicKey = ecrecover( hash, signature( hash ) );  
if ( publicKey != maker ) throw;
```

3.2.2 체결 및 부분 체결

거래소 스마트 컨트랙트는 단일 주문이 여러 번 체결되는 것을 방지하기 위해 이전에 체결된 각 주문에 대한 참조를 저장한다. 이러한 참조는 매핑 내에 저장되며, 이 경우 매핑은 32바이트 데이터 덩어리를 256비트 부호 없는 정수로 매핑하는 데이터 구조이다. 주문과 관련된 파라미터를 Keccak SHA3 함수에 전달하면, 해당 주문을 고유하게 식별할 수 있는 고유한 32바이트 해시가 생성된다(서로 다른 두 주문이 동일한 해시를 갖는 해시 충돌 가능성은 사실상 0에 가깝다). 주문이 체결될 때마다 매핑은 주문 해시와 누적 체결 값을 저장한다.

다.

테이커는 거래소 스마트 컨트랙트의 `fill` 함수를 호출할 때 추가 인자인 `valueFill`을 지정함으로써 주문을 부분적으로 체결할 수 있다. 부분 체결은 부분 체결의 합이 주문의 총 값보다 크지 않는 한, 단일 주문에 대해 여러 번 실행될 수 있다.

표 3: 테이커는 주문을 체결하려 할 때 추가 인자를 제공해야 한다.

이름	데이터 타입	설명
<code>valueFill</code>	<code>uint256</code>	체결할 <code>tokenA</code> 의 총 단위 ($valueFill \leq valueA$).

3.2.3 만료 시간

주문의 만료 시간은 주문이 서명될 때 메이커에 의해 지정된다. 만료 시간은 유닉스 에포크 이후 절대 초 단위를 나타내는 부호 없는 정수 값이다. 이 값은 서명된 이후에는 변경될 수 없다.

이더리움 가상 머신 내의 시간은 새로운 블록이 채굴될 때마다 설정되는 블록 타임스탬프에 의해 제공된다. 따라서 주문의 만료 상태는 테이커가 주문 체결 의사를 브로드캐스트하는 시간에 의존하지 않고, 채굴자가 EVM에서 `fill` 함수를 실행하는 시간에 의존한다. 채굴자는 현재 블록의 블록 타임스탬프를 이전 블록의 타임스탬프보다 이르게 설정할 수 없다.

3.2.4 주문 취소

체결되지 않았고 만료되지 않은 주문은 관련 메이커가 거래소 스마트 컨트랙트의 `cancel` 함수를 통해 취소할 수 있다. `cancel` 함수는 주문의 해시를 주문의 최대 값(`valueA`)에 매핑하여 이후 체결을 방지한다. 주문 취소는 가스 비용이 들며, 따라서 `cancel` 함수는 주로 예비 수단으로만 사용된다. 일반적으로 메이커는 주문 만료 시간을 자신이 주문을 업데이트하려는 빈도에 맞게 설정하여 온체인 거래를 피할 것으로 기대된다.

이 접근 방식의 한 가지 문제는 메이커가 주문을 취소하려는 시도와 테이커가 같은 주문을 체결하려는 시도가 거의 동시에 발생할 수 있다는 점이다. 두 당사자의 거래 중 하나는 채굴되는 순서에 따라 실패하게 되며, 이는 가스를 낭비하게 만든다. 거래가 채굴되는 순서에 대한 불확실성은 때때로 바람직하지 않은 결과로 이어질 수 있다. 이러한 불확실성은 이더리움 블록체인에 대기 중인 거래가 크게 누적될 경우 더욱 증가할 수 있다.

4. 프로토콜 토큰

크립토이코노믹 프로토콜은 합리적인 경제 주체들의 네트워크가 특정 프로세스를 완수하도록 행동을 조정하게 만드는 재정적 인센티브를 창출한다 [4,17,18]. `0x`는 본질적으로 구매자와 판매자 간 신호 전달을 촉진하기 위한 네트워크 프로토콜이지(크립토이코노믹 프로토콜이라기보다는), 교환 기능을 포함하는 dApp을 위한 오픈 표준으로 활용되도록 의도되었다. 오픈 표준을 수립하고 유지하는 것은 모든 참여자에게 운영적 부담을 더하는 조정 문제이며, 각 참여자가 서로 다른 필요와 재정적 인센티브를 가질 경우 조정은 특히 어려울

수 있다. 프로토콜 토큰은 재정적 인센티브를 일치시키고 단일 기술 표준을 중심으로 여러 당사자를 조직하는데 따른 비용을 상쇄할 수 있다. 채택을 둘러싼 인센티브를 조정하는 것이 유용할 수 있지만, 프로토콜 토큰은 훨씬 더 도전적인 문제, 즉 탈중앙화 거버넌스를 통해 변경 불가능한 스마트 컨트랙트 시스템 내에 구현된 프로토콜의 미래 적합성(future-proofing)을 보장하는 데 사용될 수 있다.

4.1 탈중앙화 거버넌스

4.1.1 지속적 통합

이더리움 스마트 컨트랙트가 블록체인에 배포되면 그 내부 로직은 변경될 수 없다. 따라서 프로토콜을 업데이트하려면 완전히 새로운 스마트 컨트랙트를 배포해야 하며, 이는 네트워크를 포크하거나 사용자가 최신 버전으로 “옵트인(opt-in)”할 때까지 해당 프로토콜에 의존하는 사용자와 프로세스를 방해하게 된다. 거래의 맥락에서 보면, 방해적인 프로토콜 업데이트는 모든 미체결 주문을 무효화할 수 있으며 각 시장 참여자가 거래 잔액에 접근하기 위해 새로운 스마트 컨트랙트를 승인하도록 요구할 수 있다. 대안적으로, 프로토콜은 두 개의 병렬로 작동하는 버전으로 포크될 수 있으며, 이는 dApp 상호운용성에 의해 창출된 네트워크 효과를 무효화할 수 있다. 스마트 컨트랙트 추상화를 통해 상위 프로세스를 방해하지 않고 프로토콜에 업데이트를 지속적으로 통합할 수 있지만, 이러한 업데이트 메커니즘은 최종 사용자에게 상당한 보안 위험을 초래할 수도 있다(최악의 경우 공격자가 사용자 자금을 접근할 수 있다). 프로토콜 토큰은 업데이트를 프로토콜에 지속적으로 통합하면서도 사용자와 이해관계자를 보호할 수 있는 탈중앙화된 업데이트 메커니즘을 구동하는 데 사용될 수 있다.

0x는 이더리움 블록체인에 배포되며, 고정된 공급량의 프로토콜 토큰이 파트너 dApp과 미래의 최종 사용자에게 발행될 것이다. 프로토콜 토큰은 두 가지 용도를 가진다: 시장 참여자가 릴레이어에게 거래 수수료를 지불하는 용도와, 프로토콜 업데이트에 대한 탈중앙화 거버넌스 용도이다. 탈중앙화 거버넌스는 그림 4에 나타난 절차에 따라 업데이트를 0x 프로토콜에 안전하게 통합하는 데 사용될 것이다. 초기에는 간단한 멀티시그 컨트랙트가 탈중앙화 거버넌스에 사용되며, 이후 더 정교한 DAO가 개발될 때까지 유지된다. 0x 프로토콜과 그 네이티브 토큰은 사용자에게 불필요한 비용을 부과하거나, 지대를 추구하거나, 릴레이어로부터 가치를 추출하지 않는다. 프로토콜의 스마트 컨트랙트는 공개적으로 접근 가능하며 완전히 무료로 사용할 수 있다. 어떤 집단이 다른 집단의 희생을 통해 이익을 얻도록 하는 메커니즘은 마련되지 않을 것이다.

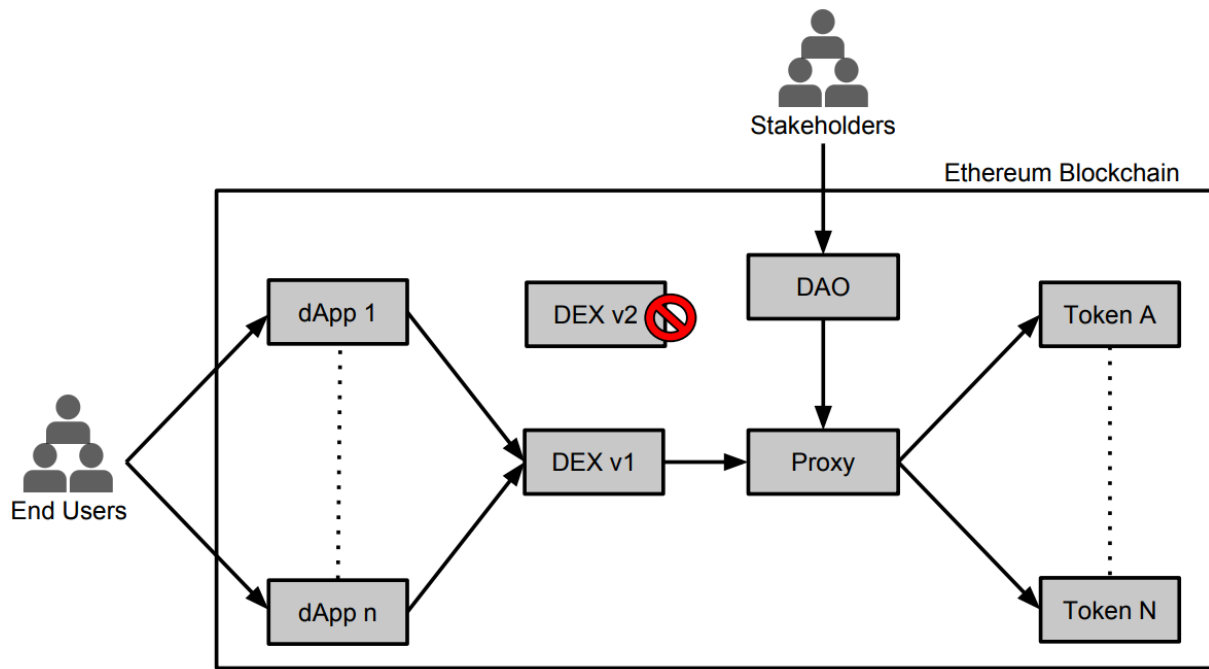


그림 4: 프로토콜 업데이트는 컨트랙트 추상화와 탈중앙화 거버넌스의 조합을 통해 네트워크를 방해하지 않고 배포될 수 있다. 최종 사용자는 자신이 거래할 토큰에 접근할 수 있도록 Proxy 컨트랙트에 권한을 부여한다. 이해관계자는 DAO를 통해 새로운 스마트 컨트랙트(DEX v2)에 구현될 프로토콜 개선안을 제안하고 선출한다. DAO는 새로운 스마트 컨트랙트를 Proxy 컨트랙트의 화이트리스트에 추가하여 사용자 토큰에 접근하도록 승인하고, 결국 프로토콜의 더 이상 사용되지 않는 버전을 리스트에서 제거한다.

4.1.2 토큰 레지스트리

주문은 기계가 읽을 수 있는 16진수 바이트코드로 구성되지만, 사람이 시각적으로 해석하기는 반드시 쉽지 않다. 토큰 레지스트리⁴ 컨트랙트는 각 ERC20 토큰에 대한 메타데이터(이름, 심볼, 컨트랙트 주소, 토큰의 최소 단위를 표현하는 데 필요한 소수 자릿수(환율 결정에 필요))와 함께 ERC20 토큰 목록을 저장하는 데 사용될 것이다. 레지스트리는 거래가 실행되기 전에 시장 참여자가 토큰 주소와 환율을 독립적으로 검증하는 데 사용할 수 있는 공식 온체인 참조 자료로 기능할 것이다. 토큰 레지스트리가 신뢰할 수 있는 정보 출처로 기능하게 되므로, 레지스트리에 토큰을 추가, 수정 또는 제거하려면 감독이 필요하다. 0x 이해관계자가 이러한 감독을 제공할 것이다. 토큰 레지스트리는 사용자가 주문의 무결성을 쉽게 검증할 수 있게 하지만, 0x 프로토콜은 ERC20 토큰 인터페이스를 사용하는 어떤 토큰과도 거래하는 데 사용할 수 있다.

(⁴ <https://github.com/ethereum/EIPs/issues/22>)

향후 프로토콜의 주문 포맷은 사람이 읽기 쉽게 수정될 수 있다. 토큰은 토큰의 컨트랙트 주소 대신 토큰 레지스트리에 등록된 세 글자 심볼로 식별될 수 있다. 이더리움 네임 서비스(ENS)는 메이커, 테이커, 릴레이어를 계정이나 컨트랙트 주소가 아닌 “theDunkle.eth”와 같은 사람이 읽을 수 있는 이름으로 식별하는 데 사용될 수 있다.

5. 요약

- 오프체인 주문 릴레이 + 온체인 결제 = 시장 메이커를 위한 낮은 마찰 비용 + 빠른 결제.
- 어떤 dApp도 연결할 수 있는 공개적으로 접근 가능한 스마트 컨트랙트.
- 릴레이어는 자신만의 유동성 풀을 만들고 거래량에 대해 거래 수수료를 부과할 수 있다.
- 표준화 + 분리 = 공유 프로토콜 계층 →
 - dApp 간 상호운용성을 제공
 - 유동성을 둘러싼 상호 이익적인 네트워크 효과를 창출
 - 진입 장벽을 줄이고, 시장 참여자의 비용을 낮춤
 - 중복을 제거하고, 사용자 경험과 스마트 컨트랙트 보안을 개선
- 탈중앙화 업데이트 메커니즘은 dApp이나 최종 사용자를 방해하지 않고 개선 사항이 프로토콜에 지속적으로 안전하게 통합되도록 한다.

6. 감사의 글

우리는 멘토, 조연자, 그리고 지식에 대해 따뜻하게 환영하고 관대하게 나누어 준 많은 이더리움 커뮤니티 사람들에게 감사의 뜻을 전하고자 한다. 특히 이 작업을 검토하고, 편집하며, 피드백을 제공해준 Joey Krug, Linda Xie, Fred Ehrsam에게 감사를 드린다. 또한 Joseph Chow, Martin Koppelman, Rebecca Migirov, Gustav Simonsson, Grant Hummer, Tom Ding, String Labs 팀 및 많은 이들과 같이, 실리콘밸리 이더리움 밋업에서 만난 주최자와 커뮤니티 구성원들에게도 감사를 전한다.

7. 부록

7.1 ERC20 토큰

ERC20은 이더리움 블록체인에서 토큰을 위한 표준 컨트랙트 ABI를 정의하며, 모든 유형의 디지털 자산에 대한 사실상의 표현 방식이 되었다. ERC20 토큰은 동일한 컨트랙트 인터페이스를 공유하므로 외부 컨트랙트와의 통합이 단순해진다.

핵심 ERC20 함수는 다음과 같다:

- transfer(to, value)
- balanceOf(owner)
- approve(spender, value)
- allowance(owner, spender)
- transferFrom(from, to, value)

EIP101에는 이더(ether)를 ERC20 토큰 표준을 따르도록 변경하는 제안이 포함되어 있다. 현재로서는 “래퍼(wrapper)” 스마트 컨트랙트를 ERC20 이더의 프록시로 사용할 수 있다. 참고를 위해 Maker 구현체나 Gnosis 구현체를 참조하라.

7.2 컨트랙트 ABI

EIP50은 구조체(struct)를 지원하기 위해 컨트랙트 ABI에 대한 확장을 제안한다. 이는 커뮤니티가 표준 *Order* 및 *Signature* 데이터 구조를 수립할 수 있도록 하며, 우리의 컨트랙트 인터페이스와 외부 컨트랙트와의 통합을 단순화한다.

7.3 이더리움 네임 서비스

EIP137 또는 이더리움 네임 서비스(ENS)는 “my-name.eth”와 같은 사람이 읽을 수 있는 이름을 이더리움 주소, Swarm 및/또는 IPFS 콘텐츠 해시 또는 다른 식별자를 나타낼 수 있는 기계가 읽을 수 있는 식별자로 변환하는 데 사용될 것이다. 또한 이름에 컨트랙트 ABI나 whois 정보와 같은 메타데이터를 연결하는 데에도 사용될 수 있다. ENS는 0x 프로토콜에 의해, 메이커, 테이커, 릴레이어를 이름으로 참조할 수 있는 보다 직관적인 메시지 포맷을 생성하는 데 사용될 것이다.

레퍼런스

- [1] coinmarketcap. <https://coinmarketcap.com/all/views/all/>. Accessed: 2017-02-016.
- [2] Wikipedia: Mt. Gox. <https://en.wikipedia.org/wiki/Mt.Gox>. Accessed: 2017-02-016.
- [3] A Timeline: ShapeShift Hacking Incident. <https://info.shapeshift.io/blog/2016/04/19/timeline-shapeshift-hacking-incident>. Accessed: 2017-02-016.
- [4] Will Warren. The difference between App Coins and Protocol Tokens. <https://medium.com/@willwarren89>, 2017.
- [5] Maker Market. <https://mkr.market/>. Accessed: 2017-02-01.
- [6] EtherOpt. <https://etheropt.github.io/>. Accessed: 2017-02-01.
- [7] Augur. <https://augur-dev.firebaseio.com/>. Accessed: 2017-02-01.
- [8] Intrinsically Tradable Tokens. <https://www.reddit.com/r/ethereum/>... Accessed: 2017-02-01.
- [9] Euler. <https://www.reddit.com/r/ethereum/>... Accessed: 2017-02-01.
- [10] Galia Benartzi Guy Benartzi, Eyal Hertzog. Bancor protocol: A hierarchical monetary system and the foundation of a global decentralized autonomous exchange. 2017.
- [11] Abraham Othman, David M Pennock, Daniel M Reeves, and Tuomas Sandholm. A practical liquidity-sensitive automated market maker. *ACM Transactions on Economics and Computation*, 1(3):14, 2013.
- [12] RaidEX. <http://www.raidex.io/>. Accessed: 2017-02-014.
- [13] Jeff Coleman. State Channels. <http://www.jeffcoleman.ca/state-channels/>. Accessed: 2017-02-014.

- [14] Ledger Labs: State Channels Wiki. <https://github.com/ledgerlabs/state-channels/wiki>. Accessed: 2017-02-014.
- [15] IDEX, Decentralized Capital. <http://www.idex.market/>. Accessed: 2017-02-01.
- [16] EtherDelta. <https://etherdelta.github.io/>. Accessed: 2017-02-01.
- [17] Fred Ehrsam. App Coins and the dawn of the Decentralized Business Model. <https://blog.coinbase.com>, 2016.
- [18] Fred Ehrsam. How to Raise Money on a Blockchain with a Token. <https://blog.gdax.com>, 2016.