OTP 인증방식을 활용한 블록체인 모니터링 시스템

2018. 11. 7

중부대학교 정보보호학과
담당 교수 양환석 교수님
3조 사이다 박진아
강신선
박지성
이지현

목 차

조원 편성

이 름	역 할
박진아	웹 페이지 & DB 구축, 블록체인 모니터링 시스템 구현, 프로젝트 총괄
강신선	웹 페이지 & DB 구축, OTP 시스템 구현, 웹 서버 구축
박지성	웹 페이지 & DB 구축, OTP 시스템 구현, 웹 서버 구축
이지현	웹 페이지 & DB 구축, 블록체인 모니터링 시스템 구현, 웹 서버 구축

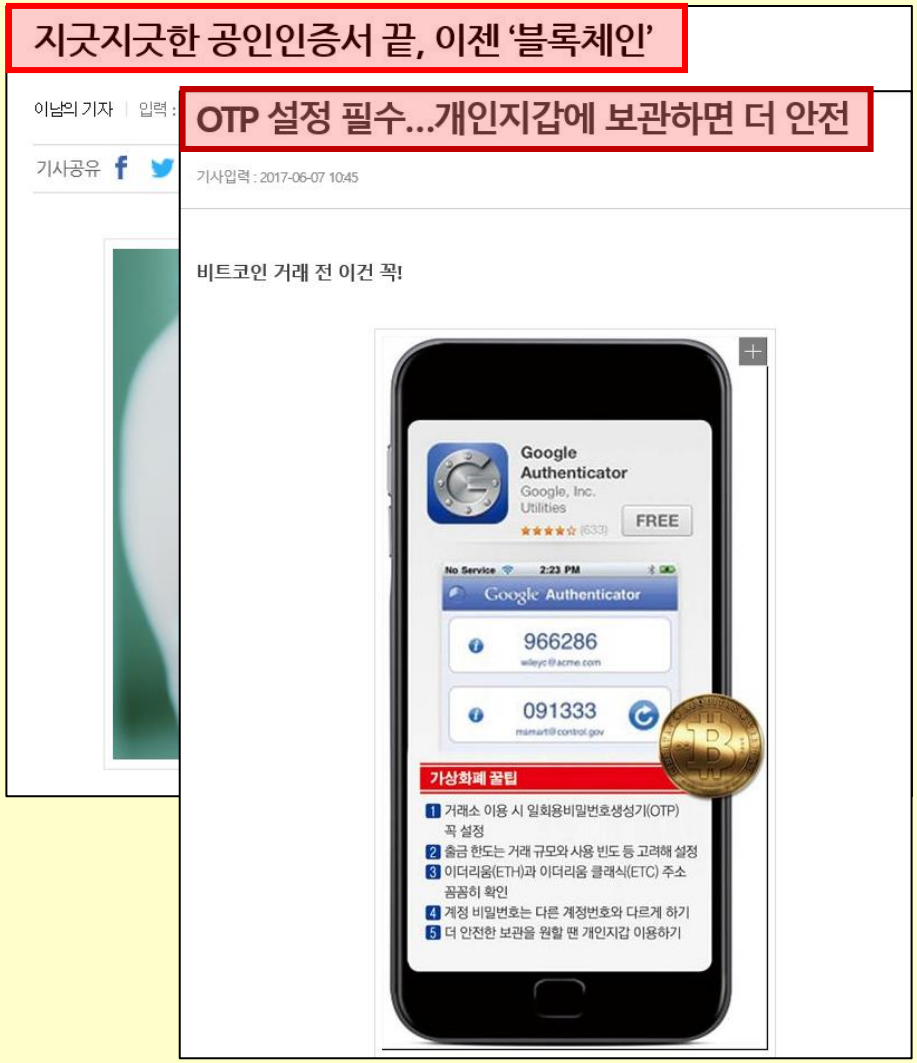
주제 선정

❒ 전자서명법 개정에 따라 공인인증서의 의무사용 규정 폐지 (2018.8.27)

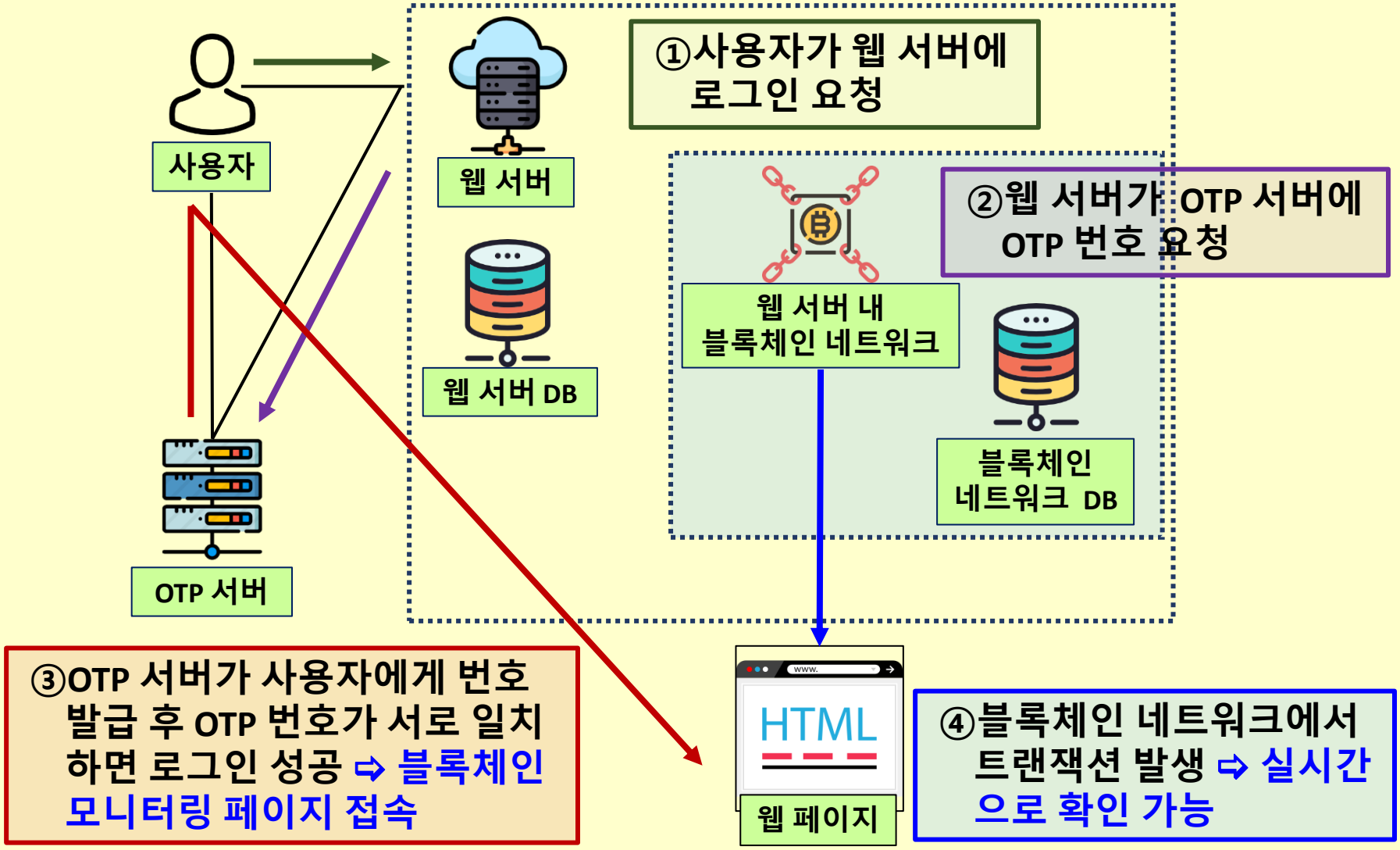
❒ 이에 따라 대체 기술 개발이 필수적

➢ 공인인증서를 대체할 인증 기술로 블록체인 기술이 주목 ⇨ 블록체인 네트워크 시스템을 연구

➢ OTP 인증을 통과한 관리자만이 블록체인 모니터링 시스템에 접근할 수 있도록 설정하여 보안성 강화



구 상 도


①사용자가 웹 서버에 로그인 요청
사용자
웹 서버
웹 서버 DB
②웹 서버가 OTP 서버에 OTP 번호 요청
웹 서버 내 블록체인 네트워크
블록체인 네트워크 DB
OTP 서버
③OTP 서버가 사용자에게 번호 발급 후 OTP 번호가 서로 일치하면 로그인 성공 ⇨ 블록체인 모니터링 페이지 접속
HTML
웹 페이지
④블록체인 네트워크에서 트랜잭션 발생 ⇨ 실시간으로 확인 가능

추진 경과

기간(2018년) / 작업	3월	4월	5월	6월	7월	8월	9월	10월
주제선정 및 자료조사								
웹 서버 구축								
웹 페이지 설계 및 구현								
OTP 구현								
블록체인 모니터링 구현								
시스템 통합 및 성능 보강								

개발 환경 및 시스템 개발(1/15)

개발 환경

서버	- Ubuntu 16.04 LTS
데이터베이스	- MySQL
개발언어	- JAVA, JSP, CSS, HTML, JavaScript, Nodejs

개발 환경 및 시스템 개발(2/15)

블록체인 종류 및 특성

퍼블릭 블록체인

1. 누구나 참여 가능
2. 누구나 거래 검증 / 승인
3. 누구나 트랜잭션 생성
4. 예) 비트코인, 이더리움

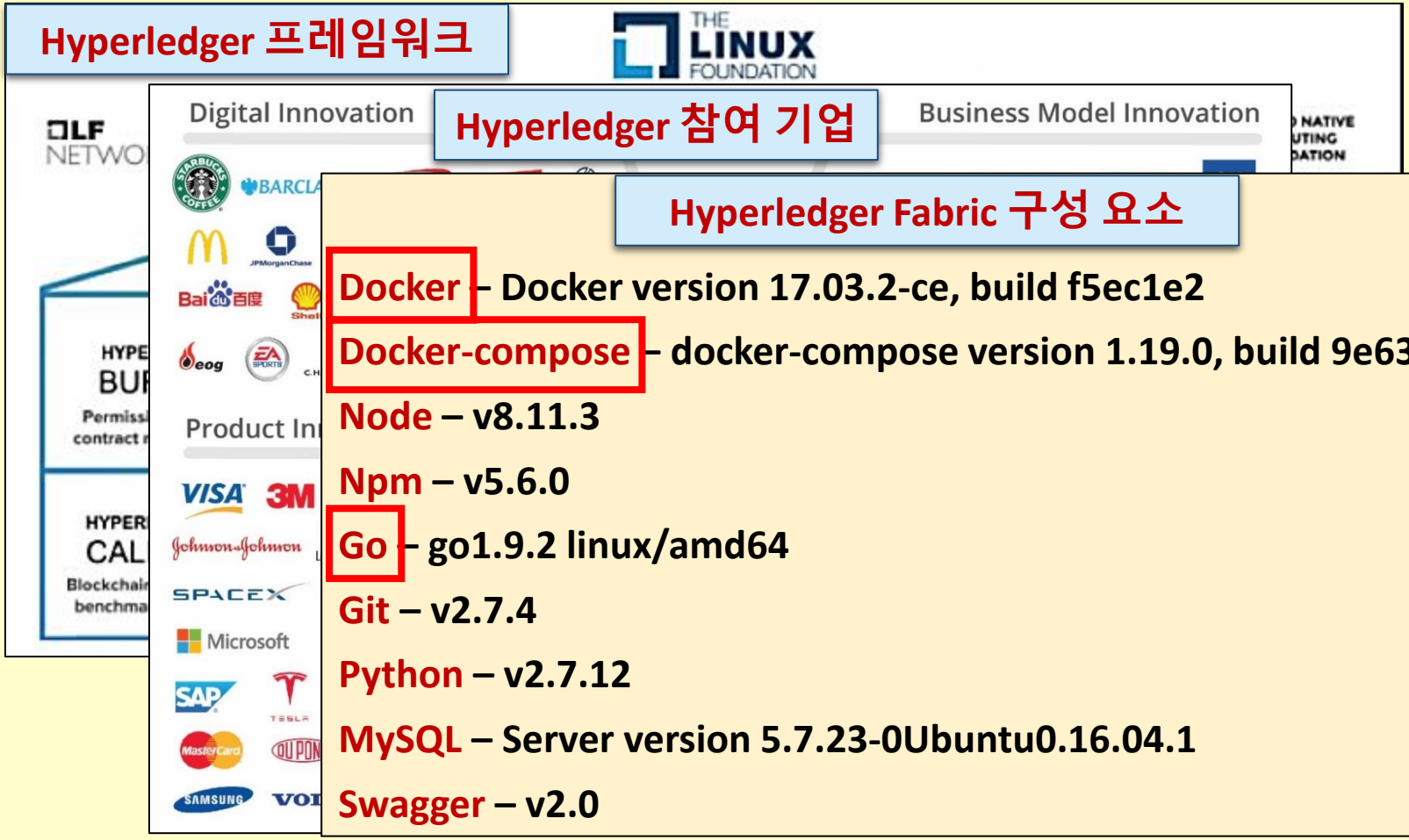
VS

프라이빗 블록체인

1. 허가된 기관, 사용자만 참여 가능
2. 허가된 기관, 사용자가 거래 검증 / 승인
3. 허가된 기관, 사용자만 트랜잭션 생성
4. 예) Hyperledger 프로젝트

개발 환경 및 시스템 개발(3/15)

Hyperledger 프로젝트



개발 환경 및 시스템 개발(4/15)

Hyperledger Fabric - docker

Docker 이미지 다운

```
-sSL https://goo.gl/6wtTN5 | bash -s 1.1.0
Installing Hyperledger Fabric docker images

===> Pulling fabric Images
==> FABRIC IMAGE: peer

===> List out hyperledger docker images
```

Docker 이미지 확인

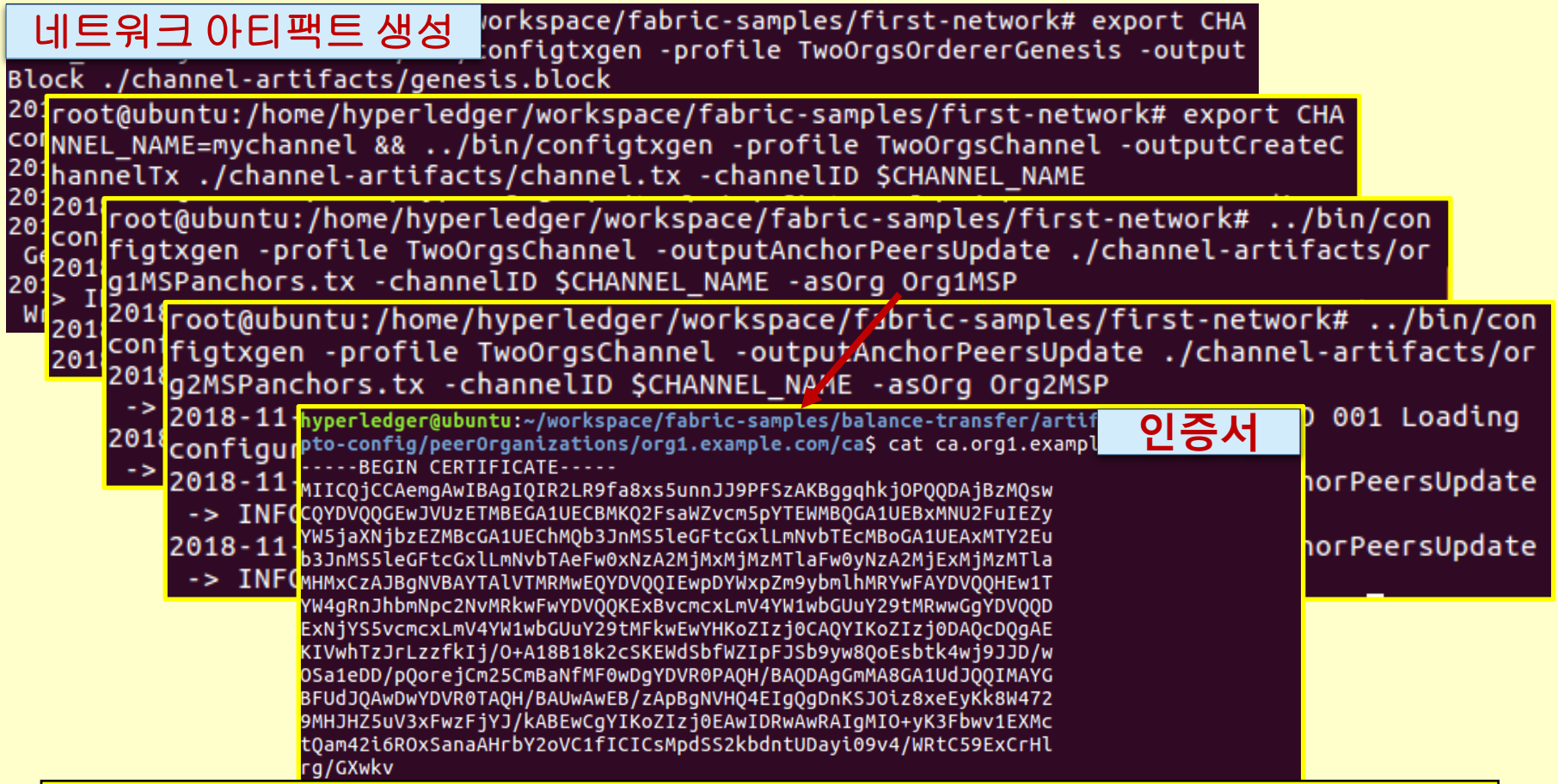
hyperledger/fabric-ca	latest	72617b4fa9b4		
hyperledger/fabric-ca	x86_64-1.1.0	72617b4fa9b4	7 weeks ago	299MB
hyperledger/fabric-tools	latest	b7bfddf508bc	7 weeks ago	1.46GB
hyperledger/fabric-tools	x86_64-1.1.0	b7bfddf508bc	7 weeks ago	1.46GB
hyperledger/fabric-orderer	latest	ce0c810df36a	7 weeks ago	180MB
hyperledger/fabric-orderer	x86_64-1.1.0	ce0c810df36a	7 weeks ago	180MB
hyperledger/fabric-peer	latest	b023f9be0771	7 weeks ago	187MB
hyperledger/fabric-peer	x86_64-1.1.0	b023f9be0771	7 weeks ago	187MB
hyperledger/fabric-javaenv	latest	82098abb1a17	7 weeks ago	1.52GB
hyperledger/fabric-javaenv	x86_64-1.1.0	82098abb1a17	7 weeks ago	1.52GB
hyperledger/fabric-ccenv	latest	c8b4909d8d46	7 weeks ago	1.39GB
hyperledger/fabric-ccenv	x86_64-1.1.0	c8b4909d8d46	7 weeks ago	1.39GB
hyperledger/fabric-zookeeper	latest	92cbb952b6f8	2 months ago	1.39GB
hyperledger/fabric-zookeeper	x86_64-0.4.6	92cbb952b6f8	2 months ago	1.39GB
hyperledger/fabric-kafka	latest	554c591b86a8	2 months ago	1.4GB
hyperledger/fabric-kafka	x86_64-0.4.6	554c591b86a8	2 months ago	1.4GB
hyperledger/fabric-couchdb	latest	7e73c828fc5b	2 months ago	1.56GB
hyperledger/fabric-couchdb	x86_64-0.4.6	7e73c828fc5b	2 months ago	1.56GB
hyperledger/fabric-tools	x86_64-1.0.0-rc1	85d6d3ca0a30	10 months ago	1.32GB

Fabric 기반의 블록체인 개발환경 구축을 위해 필요한 Docker 이미지 다운 및 확인

개발 환경 및 시스템 개발(5/15)

Hyperledger Fabric – network(1/2)

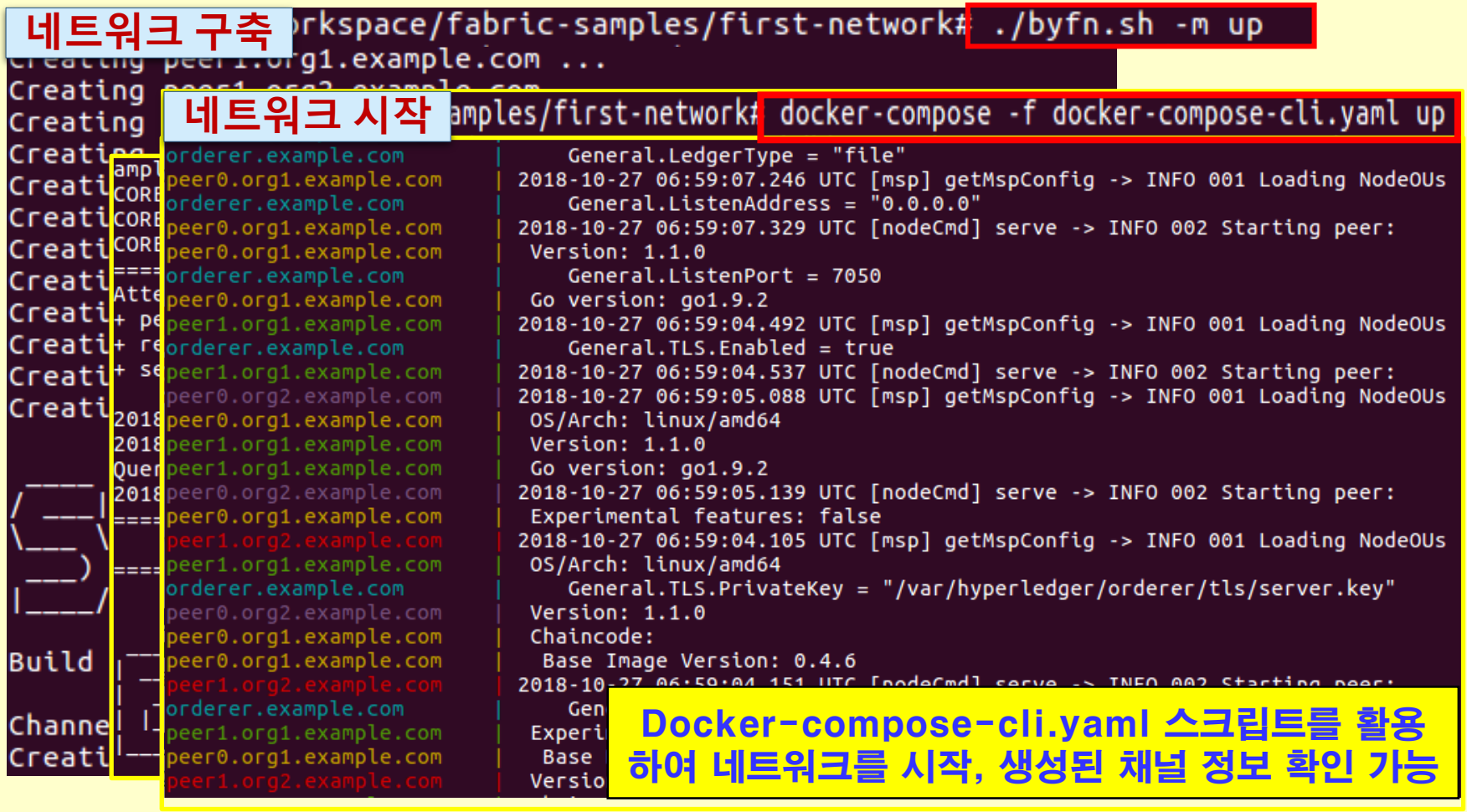
네트워크 아티팩트 생성



네트워크에 대한 인증서와 키, genesis block 및 채널 구성을 위한 트랜잭션 생성

개발 환경 및 시스템 개발(6/15)

Hyperledger Fabric – network(2/2)



개발 환경 및 시스템 개발(7/15)

Hyperledger Fabric - registration

enrollAdmin.js

```
else {
  // need to enroll it with CA server
  return fabric_ca_client.enroll({
    enrollmentID: 'admin',
    enrollm
  }).then((
    console
    return
        {us
        });
  }).then((
    admin_u
    return
```

registerUser.js

```
        return fabric_ca_client.register({enrollmentID: 'user1', affiliation: 'org1.department1',role
    }).then((secret) => {
        // next we need to enroll the user with CA server
        console.log('Successfully registered user1 - secret:'+ secret);

        return fabric_ca_client.enroll({enrollmentID: 'user1', enrollmentSecret: secret});
    }).then((enrollment) => {
      console.log('Successfully enrolled member user "user1" ');
      return fabric_client.createUser(
        {username: 'user1',
        mspid: 'Org1MSP',
        cryptoContent: { privateKeyPEM: enrollment.key.toBytes(), signedCertPEM: enrollment.certificate }
        });
    }).then((user) => {
        member_user = user;

        return fabric_client.setUserContext(member_user);
```

사용자 등록

```
root@ubun
 Store pa
Successfu
Assigned the admin user to the fabric client ::{"name":"admin","mspid":"Org1MSP","roles
":null,"affiliation":"","enrollmentSecret":"","enrollment":{"signingIdentity":"ba6c3805
290cff432a51e15f22a3958f3b9f97919ee3e6073e5da5586a58a74f","identity":{"certificate":"--
---BEGIN
AwIw\nczE
xGTAXBgNV
MDI3MDc0N
wYHKoZI\n
Dit96OpgQ
R0OBBYEFC
usalc2Xkl8\nMAoGCCqGSM49BAMCA0gAMEUCIQDzlvCyFamwO/+gRr3+F7fwxentWzd2CXW1HfKW\nHSAFzgIgE
R2l
```

```
root@ubuntu:~/workspace/fabric-samples/fabcar# node registerUser.js
 Store path:/root/workspace/fabric-samples/fabcar/hfc-key-store
Successfully loaded admin from persistence
Successfully registered user1 - secret:KDtlBFqMfIld
Successfully enrolled member user "user1"
User1 was successfully registered and enrolled and is ready to intreact with the fabric network
```

인증 서버에 관리자 등록, 관리자에 의해 인가된 사용자만 Fabric 네트워크에 참여 가능

개발 환경 및 시스템 개발(8/15)

T-OTP(Time-based One Time Password)

- 기존의 보안카드의 고정된 패스워드의 취약점을 보완하기 위하여 개발

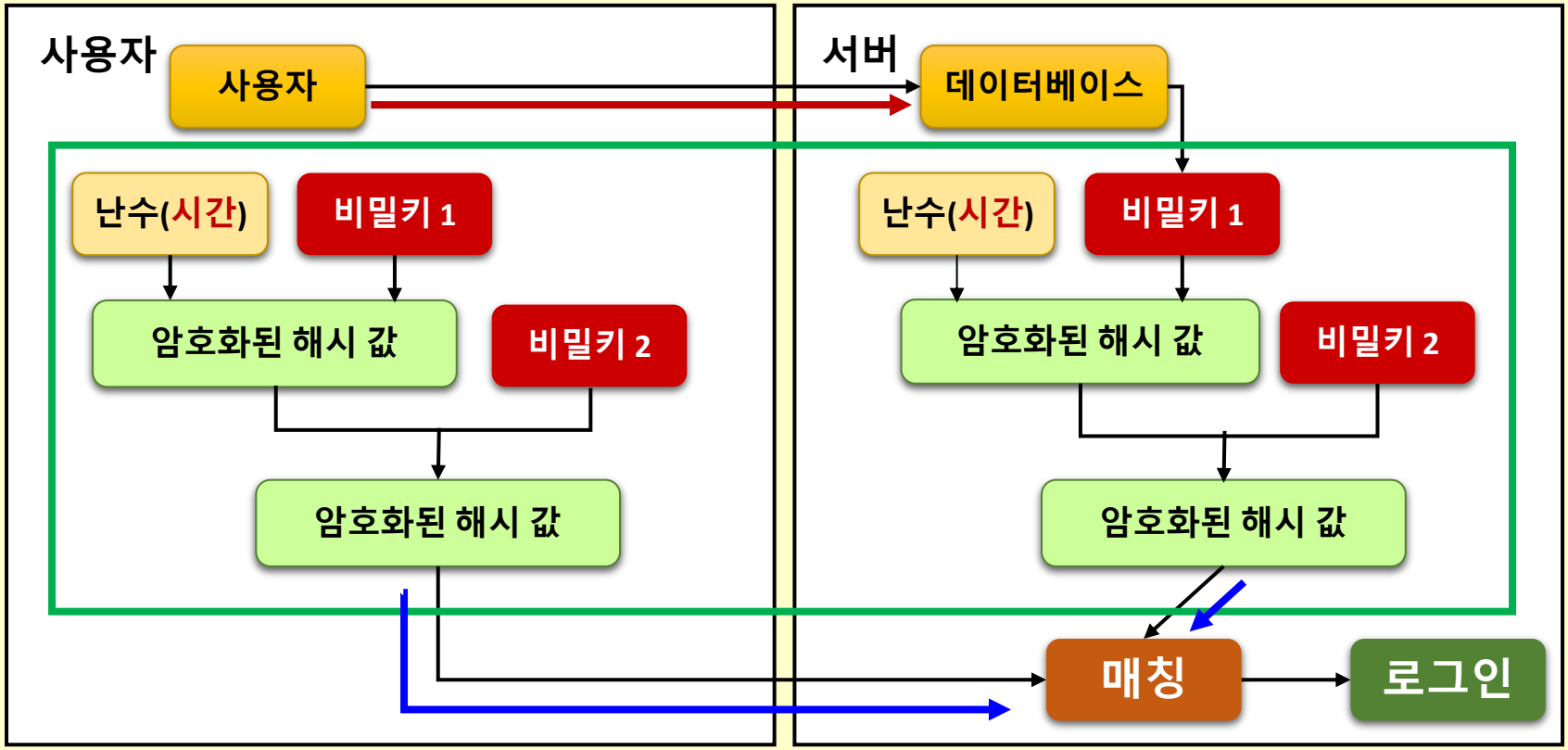
- 고정 패스워드가 아닌 설정된 시간(초)마다 다른 패스워드를 사용하는 일회용 비밀번호
  - 비밀키와 현재 시간을 결합하여 암호 생성
  - 시간에 따라 지속적으로 암호 값이 변화 (기본 30초)
  - HMAC-SHA256 알고리즘을 이용하여 암호 생성

- OTP 종류는 시간 동기화방식(T-OTP), 이벤트 방식(H-OTP) 등으로 분류

개발 환경 및 시스템 개발(9/15)

T-OTP 알고리즘

비밀키 1=id, 비밀키 2=pw,
시간을 받아서 난수를 생성



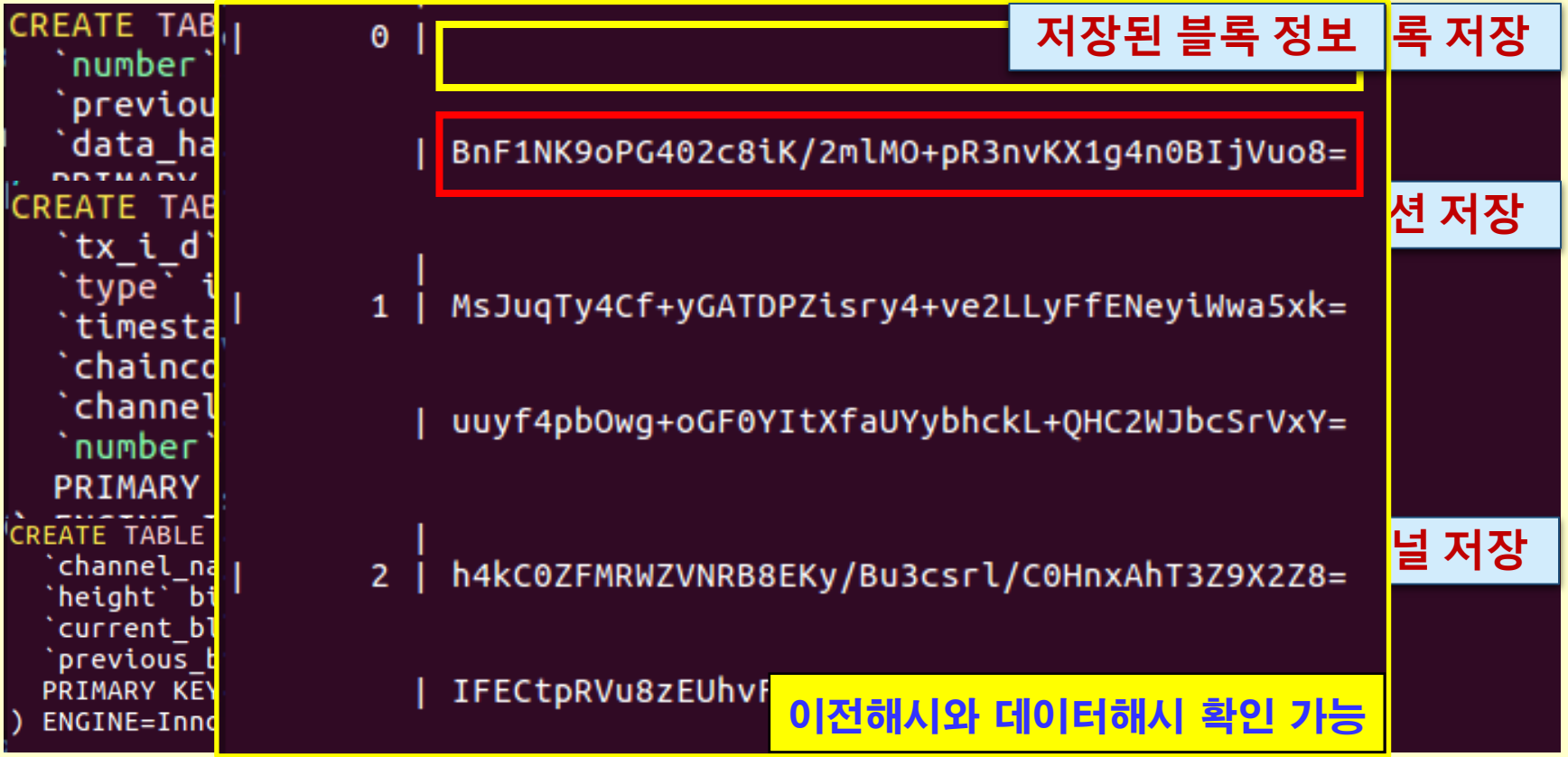
① 사용자가 웹 서버 로그인 요청

② SHA256 알고리즘을 사용, 해시 값 생성

③ 사용자와 서버의 해시 값 동일 ⇨ 로그인 성공

개발 환경 및 시스템 개발(10/15)

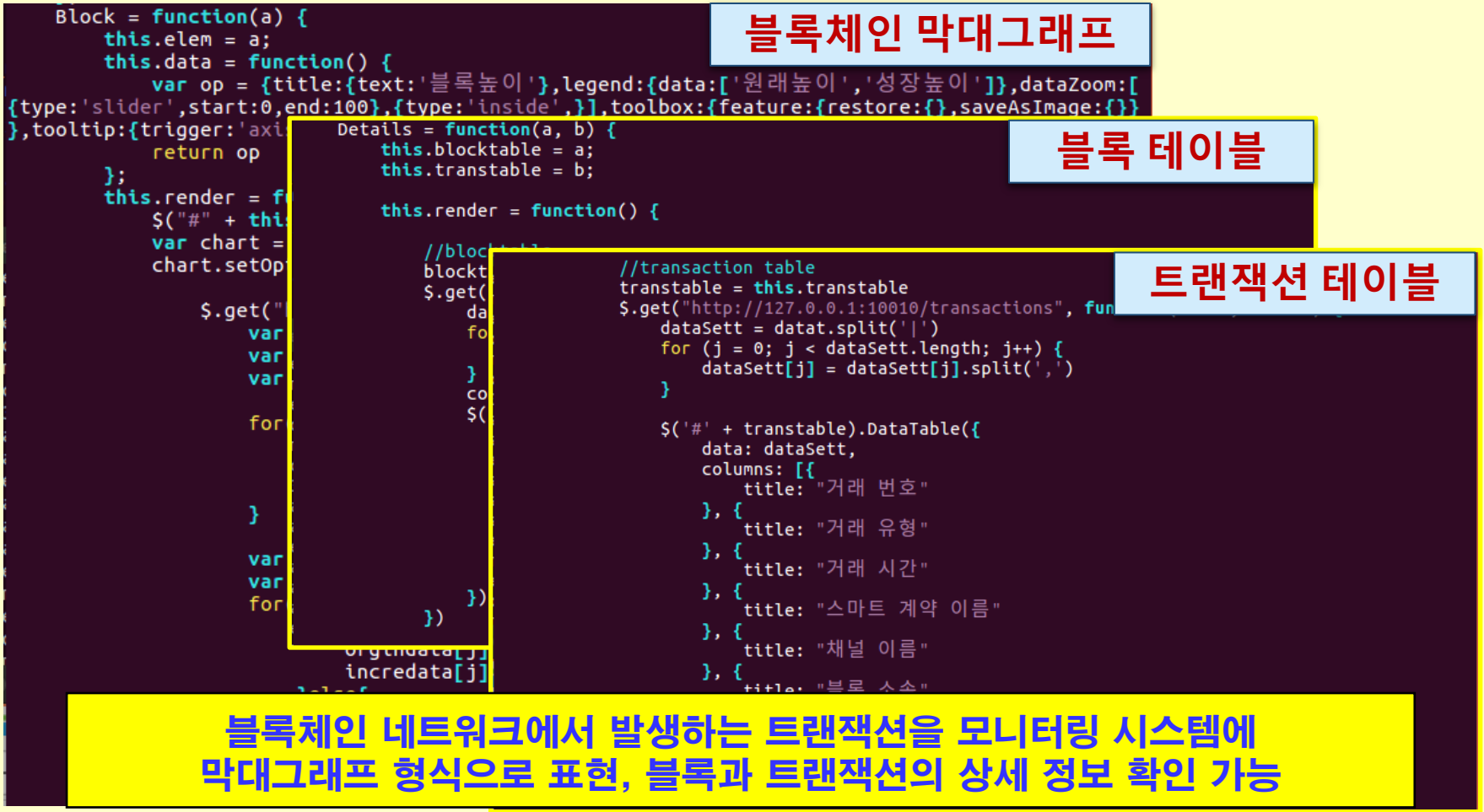
시스템 개발 (1/6)



블록, 트랜잭션,채널의 정보 등이 블록체인 DB에 저장

개발 환경 및 시스템 개발(11/15)

시스템 개발 (2/6)



블록체인 네트워크에서 발생하는 트랜잭션을 모니터링 시스템에
막대그래프 형식으로 표현, 블록과 트랜잭션의 상세 정보 확인 가능

개발 환경 및 시스템 개발(12/15)

시스템 개발 (3/6)

사용자 계정 유효성 검증

```
<script>
    function login(){
        var queryString = $("form[id=loginForm]").serialize();

        $.ajax({
            type:"POST
            url:"<%=re
            data: quer
            success :
                if(dat
                    al
                }else{
                    $(
                    wi
                }
            },error :
            }
        });
    }
</script>
```

사용자 계정 정보 확인 쿼리

```
<%@ page import = "java.sql.*" %>
<%@ page import = "java.util.Map" %>
<%@ page import = "java.util.HashMap" %>
<%@ include file="/jdbcDB.jsp" %>
<%
    request.setCharacterEncoding("UTF-8");
    String email = request.getParameter("email");
    String password = request.getParameter("password");
    Map<String, String> User = new HashMap<String, String>();
    out.clear();
    try {
        stmt = conn.createStatement();
        String sql = String.format("SELECT * from user WHERE email = '%s' AND password = '%s'", email, password);
        ResultSet rs = stmt.executeQuery(sql);
        int count = 0;
            while (rs.next()) {
            count++;
            User.put("name", rs.getString("name"));
            User.put("year", rs.getString("year"));
            User.put("month", rs.getString("month"));
            User.put("day", rs.getString("day"));
            User.put("birthDay", rs.getString("birthDay"));
            User.put("gender", rs.getString("gender"));
            User.put("idCardNumber", rs.getString("idCardNumber"));
            User.put("email", rs.getString("email"));
```

사용자가 입력한 값과 DB에 저장된 값을 비교하여 동일한 경우 로그인

개발 환경 및 시스템 개발(13/15)

시스템 개발 (4/6)

hmac 암호화 알고리즘

```
private byte[] hmac_sha(String crypto, byte[] keyBytes, byte[] text) {
    try {
        Mac hmac;
        hmac = Mac.getInstance(crypto);
        SecretKeySpec macKey = new SecretKeySpec(keyBytes, "RAW");
        hmac.init(macKey);
        return hmac.doFinal(text);
    } catch (GeneralSecurityException gse) {
        throw new Un
    }
}
```

HEX 문자열 Byte변환

```
private byte[] hexStr2Bytes(String hex) {
    byte[] bArray = new BigInteger("10" + hex, 16).toByteArray();

    byte[] ret = new byte[bArray.length - 1];
    for (int i = 0; i < ret.length; i++)
        ret[i] = bArray[i + 1];
    return ret;
}
```

HMAC (Hash-based Message Authentication Code)
암호 해시 알고리즘을 매개변수로 사용하여 T-OTP 값 생성

개발 환경 및 시스템 개발(14/15)

시스템 개발 (5/6)

T-OTP 값 생성

```
private final int[] DIGITS_POWER
```

T-OTP 값 생성

```
public String generateTOTP(String key, String time, String retur
    int codeDigits = Integer.decode(return
    String result = null;
    while (time.length() < 16)
        time = "0" + time;

    byte[] msg = hexStr2Bytes(time);
    byte[] k = hexStr2Bytes(key);
    byte[] hash = hmac_sha(crypto, k, msg)

    int offset = hash[hash.length - 1] & 0

    int binary = ((hash[offset] & 0x7f) <<
            | ((hash[offset + 2] & 0xff) <

    int otp = binary % DIGITS_POWER[codeDi

    result = Integer.toString(otp);
    while (result.length() < codeDigits) {
        result = "0" + result;
    }
    return result;
}
```

```
00000000 };
```

T-OTP 값 메일로 전송

```
Session session = Session.getDefaultInstance(props
 protected PasswordAuthentication getPasswordAuthe
  return new PasswordAuthentication(user, password
 }
}); // SMTP 사용자정보를 이용해 naver메일과 연동


try {
 MimeMessage message = new MimeMessage(session);
 message.setFrom(new InternetAddress(user));
 message.addRecipient(Message.RecipientType.TO, new InternetAddress(to));


 message.setSubject("OTP Number Send"); // 메일 제목

 String otpnum=getId(); // otp 값 받아오기

 message.setText(otpnum); // otp 값 입력
```

Key : Hex encode
returnDigits : 반환 할 자릿 수
time : 시간을 반영하는 값
crypto : 사용할 암호화 함수

개발 환경 및 시스템 개발(15/15)

시스템 개발 (6/6)

시간값 가져오기

T-OTP 해시값

```
public boolean isOK() {
    boolean ok = false;
    String seed32 = "secret";
    long nowtime =
    long T0 = 0;
    long X = 60;
    long testTime[]
    String steps =
    DateFormat df =
    df.setTimeZone(
        try {
            for (int i = 0; i < testTime.length; i++) {
                long T = (testTime[i] - T0) / X;
                steps = Long.toHexString(T).toUpperCase();
                while (steps.length() < 16)
                    steps = "0" + steps;
                if (id.equals(generateTOTP(seed32, steps, "8", "HmacSHA256")))
                    ok = true;
            }
        } catch (final Exception e) {
            System.out.println("Error : " + e);
        }
        String dab = generateTOTP(seed32, steps, "8", "HmacSHA256");
        if (dab.equals(id))
            ok = true;
        return ok;
    }
```

암호화 된 OTP 값이 일치할 경우 true, 일치하지 않을 경우 false

개발 시스템 운영(1/5)

시스템 운영 (1/5)

개발 시스템 운영(2/5)

시스템 운영 (2/5)

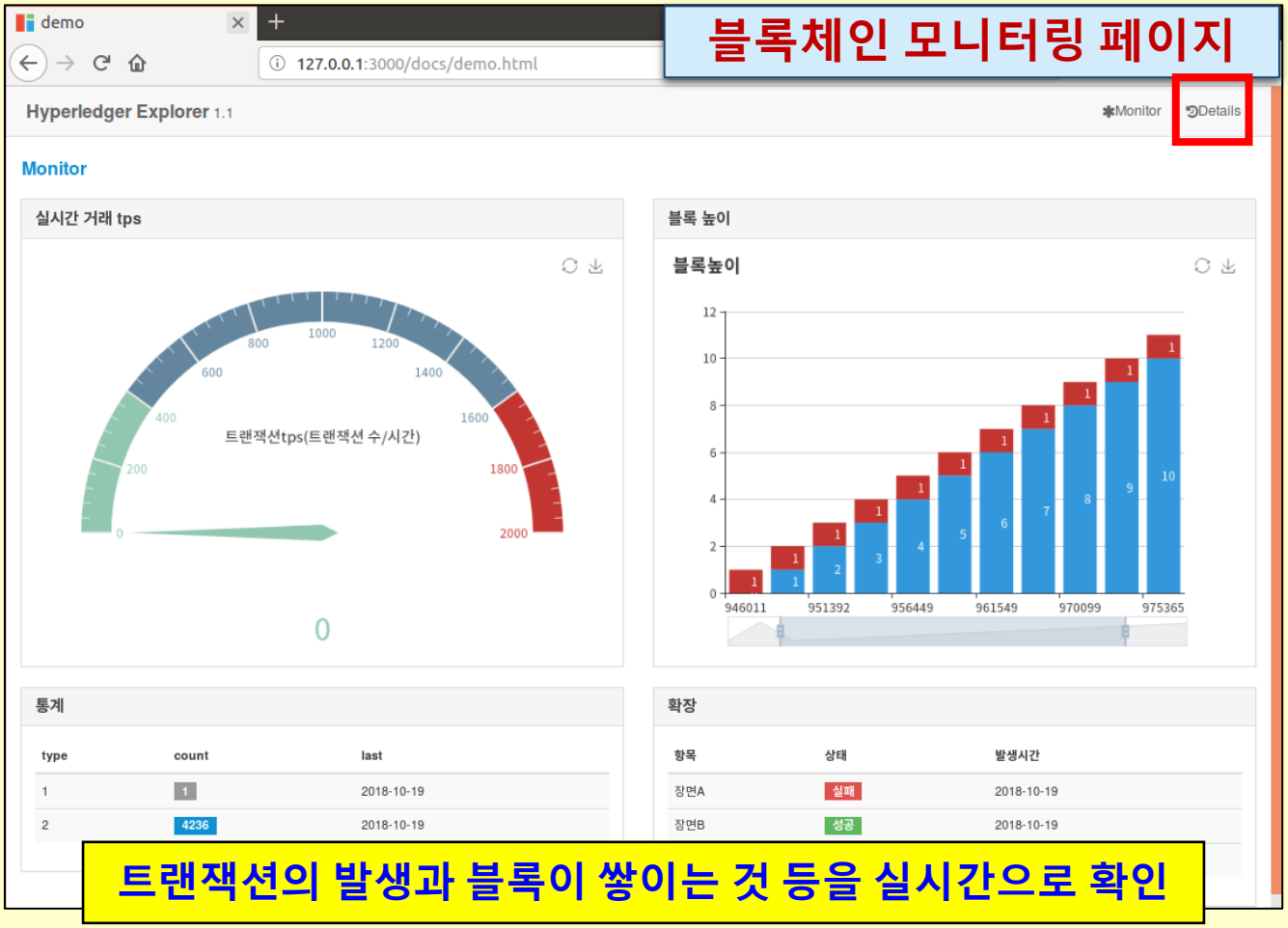
개발 시스템 운영(3/5)

시스템 운영 (3/5)

정상 로그인 후 확인 버튼 클릭시 블록체인 모니터링 시스템으로 이동

개발 시스템 운영(4/5)

시스템 운영 (4/5)



개발 시스템 운영(5/5)

시스템 운영 (5/5)

결론 및 기대효과

◆ OTP 인증 방식의 블록체인 모니터링 시스템 개발 성과

- ✓ OTP 인증을 통해 인가된 사용자만이 블록체인 모니터링 시스템에 접근
- ✓ 모니터링 시스템을 통해 사용자들의 트랜잭션과 그에 따른 블록의 변화를 실시간으로 확인 함으로써 시스템의 성능을 검증

◆ 기대효과 (소감)

- ✓ 4차 산업혁명의 핵심기술 중 하나인 블록체인 기술을 연구하면서 많은 시행착오를 겪고 성장하는 계기
- ✓ 특히 조원들이 상호 의지하며 포기하지 않고 노력한 결과가 조그만 성과로 나타난 것이 보람찬 결실

Q&A

감사합니다