对称加解密算法解析

一、概述

cryptosystem密码学系统分为私钥系统及公钥系统。

私钥系统：指加解密双方事先做了私有信息约定，采用对称密钥算法；
公钥系统：指发送方用公开凭证对数据进行加密后传输，接收方使用私有凭证进行解密，采用非对称密钥算法；

对称加密分类

流加密（stream cipher），加密和解密双方使用相同伪随机加密数据流，一般都是逐位异或或者随机置换数据内容，常见的流加密算法如RC4。
分组加密加密（block cipher），也叫块加密，将明文分成多个等长的模块（block），使用确定的算法和对称密钥对每组分别加密解密。
高级的分组加密建立以迭代的方式产生密文，每轮产生的密文都使用不同的子密钥，而子密钥生成自原始密钥。
数据加密中分组方式成为分组模式，如ECB；当加密中数据长度不足以满足分组时需要进行填充，此时采用的方式对应填充算法，如PKCS5Padding。

二、对称密钥算法

DES

Data Encryption Standard，数据加密标准，由IBM研究设计。
密钥长度8字节，有效位56bit；其中，分组为64bit=8字节。

3DES

DES像 AES过渡的加密标准。

由3个64bit的DES密钥对数据进行三次加密。

密钥长度为24字节，有效位168bit。

AES

Advanced Encryption Standard，高级加密标准。

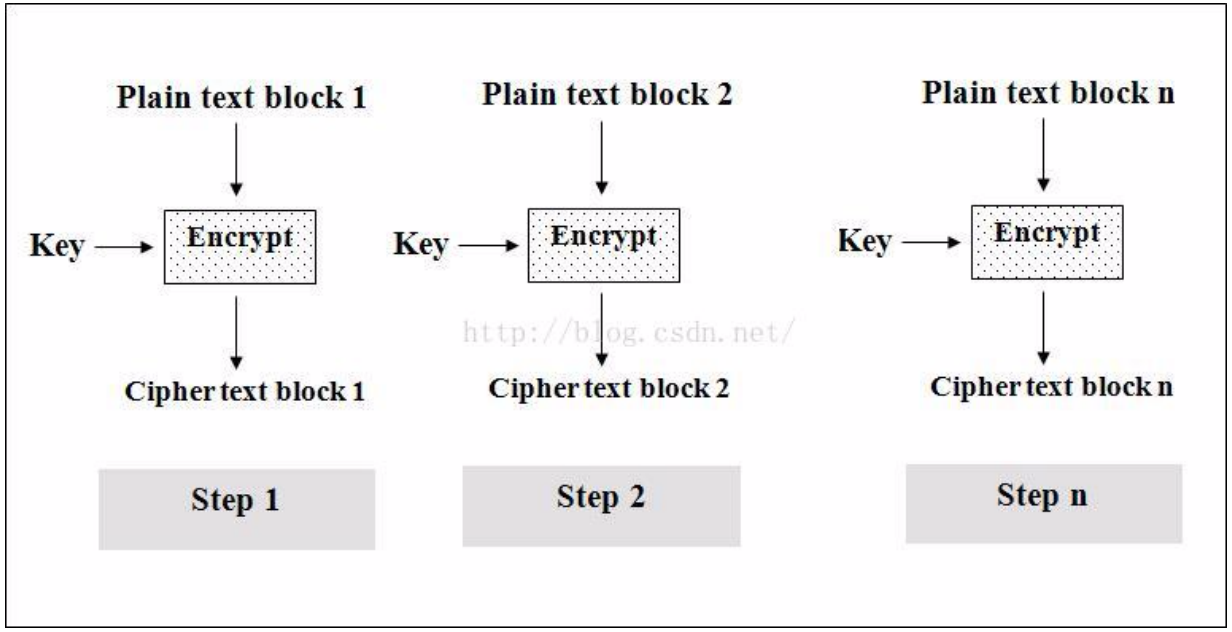
包括AES-128;AES-192;AES-256算法，分组大小为128bit=16字节。

三、密码分组模式

1 ECB

Electronic Code Book，电码本模式

相同分组输出相同的密钥，简单且利于并行运算，但无法隐藏模式，也容易招致攻击



搜索

找找看

常用链接
我的随笔
我的评论
我的参与
最新评论
我的标签

随笔分类
0.JAVA技术(39)
1.架构设计(3)
2.安全技术(9)
3.前端技术(2)
4.测试技术(2)
5.数据库中间件(21)
7.工具技巧(9)
8.构建技术(5)
9.基础原理(2)
O.开放平台(3)
P.行业相关(1)
S.敏捷管理
Z.心得杂谈(9)

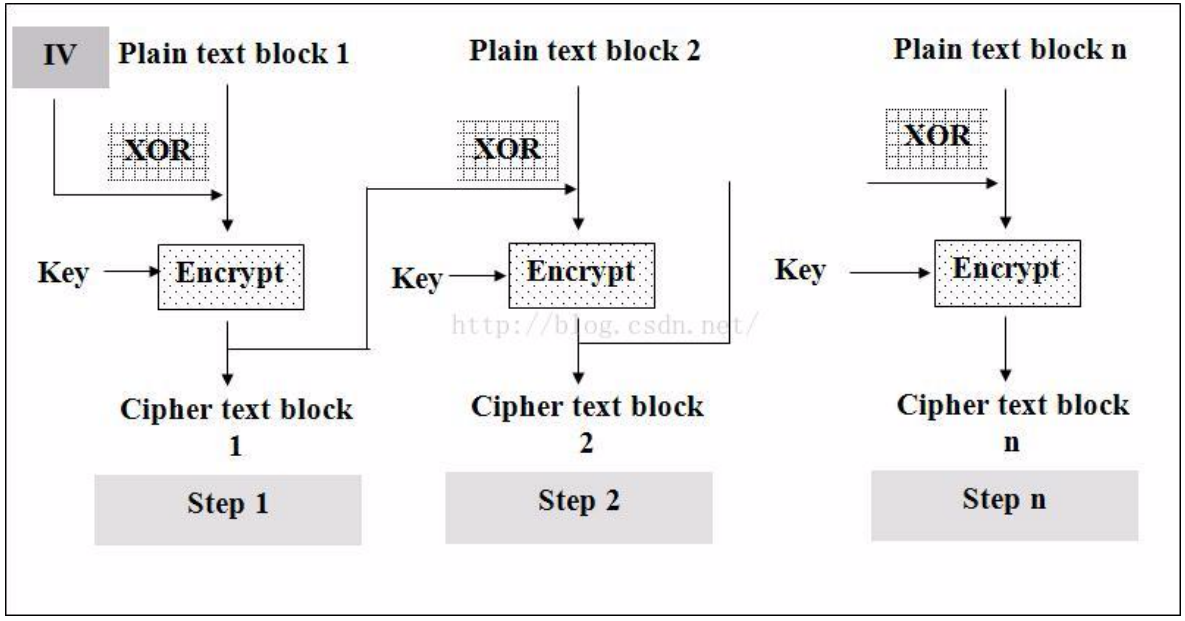
随笔档案
2019年5月 (2)
2019年4月 (4)
2019年3月 (7)
2019年2月 (2)
2018年12月 (2)
2018年11月 (6)
2018年9月 (3)

2 CBC

Cipher Block Chaining，密文分组链模式

需要初始化向量IV(长度与分组大小相同)，第一组的密文与第二组数据XOR计算后再进行加密产生第二组密文

安全性较好，TLS、IPSec等标准的推荐模式，但不利于并行运算



3 CFB

links

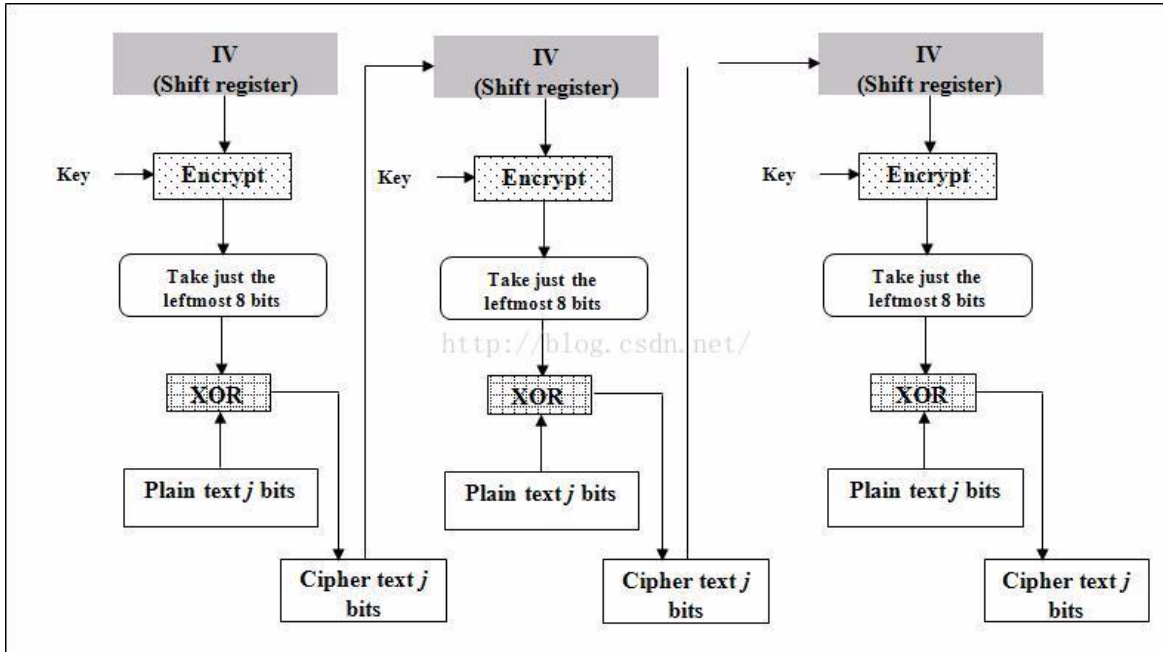
ascii 图表

ascii艺术字

最新评论

1. Re:Java条形码生成技术-Barcode4j

Cipher Feedback，密文反馈模式



4 OFB

怎么隐藏条形码下方的文本?或者设置为自定义文本?

--习惯沉淀

2. Re:成为高手前必懂的TCP干货

@海向

--美码师

3. Re:成为高手前必懂的TCP干货

6

--海向

4. Re:redis通过pipeline提升吞吐量

@ericlfredis-stat ，可参考这里的：...

--美码师

5. Re:redis通过pipeline提升吞吐量

楼主，请教下性能测评是使用的什么工具？3Q！

--ericlf

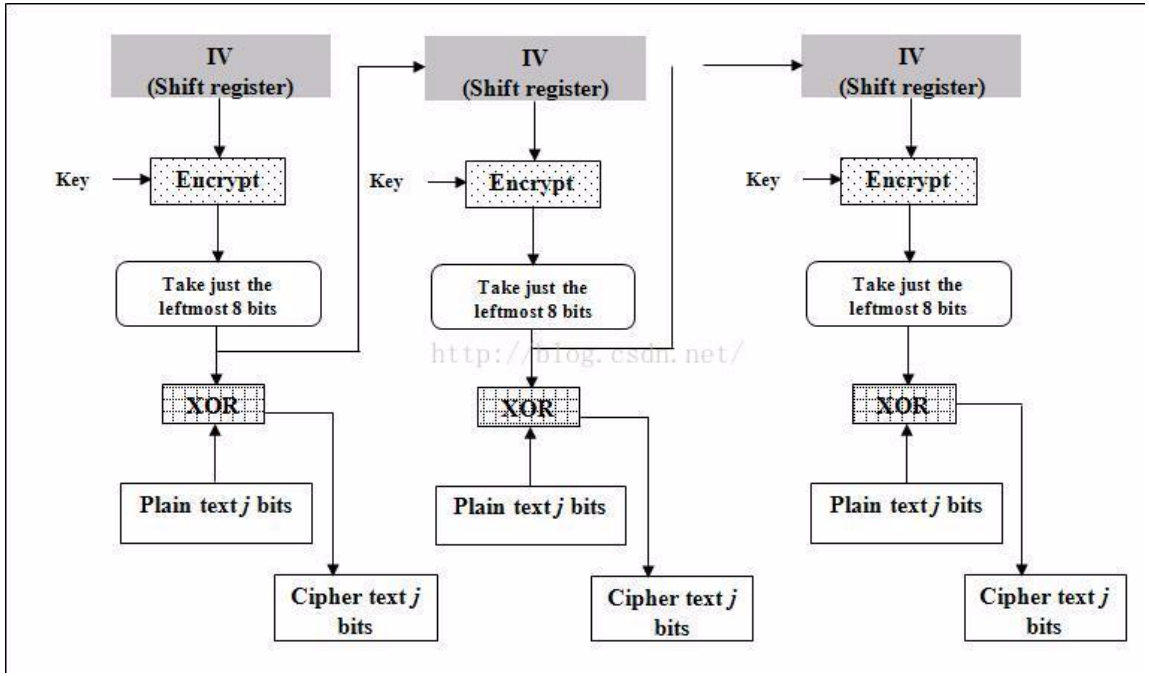
阅读排行榜

1. 使用 openssl 生成证书(62415)
2. mysql 索引过长1071-max key length is 767 byte(56645)
3. Java条形码生成技术-Barcode4j(36608)
4. MQTT服务器搭建-mosquitto1.4.4安装指南(26266)
5. 使用keytool 生成证书(18705)

评论排行榜

1. 情人节，送女友一桶代码可否？(36)
2. 软能力那点事，你知多少(18)
3. 老兵的十年职场之路(一)(9)
4. redis通过pipeline提升吞吐量(6)
5. MQTT服务器搭建-mosquitto1.4.4安装指南(6)

Output Feedback (OFB)，输出反馈模式



三、填充算法

1 NoPadding，无填充算法，通常要求数据满足分组长度要求；

2 ZerosPadding，全部填充为0；

3 PKCS5Padding，填充字节数；

4 others…

DES像 AES过渡的加密标准

由3个64bit的DES密钥对数据进行三次加密

密钥长度为24字节，有效位168bit

四、代码示例

```
/**
 * 加密工具类
 *
 * <pre>
 * AES支持128/192/256,取决于密钥长度(与位数对应)
 * DES密钥长度8字节
 * 3DES密钥长度24字节
 *
 * 采用CBC 需指定初始向量IV，长度与分组大小相同
 * DES为8字节；AES为16字节
 *
 * </pre>
 */
public class Crypto {
    static {
        // add bouncycastle support for md4 etc..
        Security.addProvider(new BouncyCastleProvider());
    }
    public static enum CryptType {
        DES_ECB_PKCS5("DES/ECB/PKCS5Padding"),
        DES_CBC_PKCS5("DES/CBC/PKCS5Padding", 8),
        DESede_ECB_PKCS5("DESede/ECB/PKCS5Padding"),
        DESede_CBC_PKCS5("DESede/CBC/PKCS5Padding", 8),
        AES_ECB_PKCS5("AES/CBC/PKCS5Padding", 16),
        AES_CBC_PKCS5("AES/CBC/PKCS5Padding", 16),
        AES_CBC_PKCS7("AES/CBC/PKCS7Padding", 16);
        public final String algorithm;
        public final String keyAlg;
        public final int ivlen;
        private CryptType(String algorithm, int ivlen) {
            this.algorithm = algorithm;
```

```
            this.keyAlg = this.algorithm.substring(0, this.algorithm.indexOf('/'));
            this.ivlen = ivlen;
        }
        private CryptType(String algorithm) {
            this(algorithm, 0);
        }
        @Override
        public String toString() {
            return this.algorithm;
        }
    }
    /**
     * Initialize the key
     *
     * @param type
     * @return
     */
    public static String initKey(CryptType type) {
        try {
            KeyGenerator generator = KeyGenerator.getInstance(type.keyAlg);
            SecretKey secretKey = generator.generateKey();
            byte[] key = secretKey.getEncoded();
            return Codec.byteToHexString(key);
        } catch (Exception e) {
            throw new RuntimeException(e);
        }
    }
    /**
     * generate default ivparam for type
     *
     * @return
     */
```

```
    public static byte[] generateDefaultIv(CryptType type) {
        byte[] iv = new byte[type.ivlen];
        for (int i = 0; i < iv.length; i++) {
            iv[i] = 0x01;
        }
        return iv;
    }
    /**
     * Encrypt the value with the encryption standard.
     *
     * @param value
     *            raw string
     * @param key
     *            in hex format
     * @param iv
     *            in hex format if exist
     * @param type
     * @return result in hex format
     */
    public static String encrypt(String value, String key, String iv, CryptType
type) {
        byte[] dvalue;
        try {
            dvalue = value.getBytes("utf-8");
        } catch (UnsupportedEncodingException e) {
            throw new RuntimeException(e);
        }
        byte[] dkey = Codec.hexStringToByte(key);
        byte[] div = null;
        if (iv != null && iv.length() > 0) {
            div = Codec.hexStringToByte(iv);
        }
```

```
        byte[] result = encrypt(dvalue, dkey, div, type);
        return Codec.byteToHexString(result);
    }
    /**
     * Encrypt the value with the encryption standard.
     *
     * <pre>
     * key must have the corresponding length.
     *
     * if use cbc mode which need iv param, the iv must not be null,
     * and iv data length is 16 for aes, 8 for des
     *
     * </pre>
     *
     * @param value
     * @param key
     * @param iv
     * @return
     */
    public static byte[] encrypt(byte[] value, byte[] key, byte[] iv, CryptType
type) {
        try {
            SecretKeySpec skeySpec = new SecretKeySpec(key, type.keyAlg);
            Cipher cipher = Cipher.getInstance(type.algorithm);
            IvParameterSpec ivparamSpec = null;
            if (iv != null) {
                ivparamSpec = new IvParameterSpec(iv);
            }
            cipher.init(Cipher.ENCRYPT_MODE, skeySpec, ivparamSpec);
            return cipher.doFinal(value);
        } catch (Exception ex) {
            throw new RuntimeException(ex);
```

```
        }
    }
    /**
     * Encrypt the value with the encryption standard.
     *
     * @param value
     *            encoded data in hex format
     * @param key
     *            in hex format
     * @param iv
     *            in hex format if exist
     * @param type
     * @return result raw string
     */
    public static String decrypt(String value, String key, String iv, CryptType
type) {
        byte[] dvalue = Codec.hexStringToByte(value);
        byte[] dkey = Codec.hexStringToByte(key);
        byte[] div = null;
        if (iv != null && iv.length() > 0) {
            div = Codec.hexStringToByte(iv);
        }
        byte[] result = decrypt(dvalue, dkey, div, type);
        try {
            return new String(result, "utf-8");
        } catch (UnsupportedEncodingException e) {
            throw new RuntimeException(e);
        }
    }
    /**
     * Decrypt the value with the encryption standard.
     *
```

```
     * <pre>
     * key must have the corresponding length.
     *
     * if use cbc mode which need iv param, the iv must not be null,
     * and iv data length is 16 for aes, 8 for des
     *
     * </pre>
     *
     * @param value
     * @param key
     * @param iv
     * @param type
     * @return
     */
    public static byte[] decrypt(byte[] value, byte[] key, byte[] iv, CryptType
type) {
        try {
            SecretKeySpec skeySpec = new SecretKeySpec(key, type.keyAlg);
            Cipher cipher = Cipher.getInstance(type.algorithm);
            IvParameterSpec ivparamSpec = null;
            if (iv != null) {
                ivparamSpec = new IvParameterSpec(iv);
            }
            cipher.init(Cipher.DECRYPT_MODE, skeySpec, ivparamSpec);
            return cipher.doFinal(value);
        } catch (Exception ex) {
            throw new RuntimeException(ex);
        }
    }
}
```

key 长度受限问题

```
Exception in thread "main" java.security.InvalidKeyException: Illegal key size or default parameters
```

问题原因：因软件出版政策原因，默认 jdk 环境做了限制，当AES加密密钥大于128位时，会出现以上异常；

解决办法：下载JCE扩展，替换至 ${java_home}/jre/lib/security

http://www.oracle.com/technetwork/java/javase/downloads/index.html

五、参考文档：

http://m.blog.csdn.net/article/details?id=51066799

http://www.blogjava.net/amigoxie/archive/2014/07/06/415503.html

分类：2.安全技术

标签：安全


美码师
关注 - 7
粉丝 - 90
+加关注

« 上一篇：数字信息摘要常见算法
» 下一篇：linux vim 无权限保存解决办法

posted @ 2016-12-19 00:22 美码师 阅读(1995) 评论(0) 编辑 收藏

相关博文：
· 对称和非对称加解密
· java 加解密实例（对称——非对称）
· Cryptography中的对称密钥加解密：fernet算法探究

· 对称密码体制和非对称密码体制
· JAVA加解密 -- 对称加密算法与非对称加密算法

最新新闻:
· 鱼在水中也憋气
· 谷歌宣布即将淘汰32位版Android Studio与Android模拟器
· Mozilla 正式为 Firefox 推出全新 logo
· 天文学家称月球最大的陨石坑下方隐藏着神秘物质
· 可循环利用食品包装透明薄膜问世
» 更多新闻...