计算机科学
COMPUTER SCIENCE

一种类自同步ZUC算法的认证加密方案

徐睿, 彭长根, 许德权

引用本文

徐睿, 彭长根, 许德权. 一种类自同步ZUC算法的认证加密方案[J]. 计算机科学, 2023, 50(10): 377-382.

XU Rui, PENG Changgen, XU Dequan. Authenticated Encryption Scheme of Self-synchronous-like ZUC Algorithm [J]. Computer Science, 2023, 50(10): 377-382.

计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.220800007

一种类自同步 ZUC 算法的认证加密方案

徐 睿 彭长根 许德权
贵州大学计算机科学与技术学院 贵阳 550025
省部共建公共大数据国家重点实验室 贵阳 550025
(xurui_xr96@163.com)

摘 要 针对国密 ZUC 算法的认证加密的安全性、效率以及轻量化需求,提出了一种类自同步 ZUC 的关联数据认证加密方案 ZUCAE。该方案通过改进祖冲之流密码算法(ZUC-256)的 LFSR 层算法,设计实现了一种类似自同步流密码的 ZUC-SSL 算法,利用该算法使密文参与到状态更新函数中,用于认证码的生成。ZUC-256 算法进行消息加密,通过优化初始化模块,将关联数据嵌入到初始化过程中,实现了密钥流生成和加密并行进行,解密前进行消息认证,减少计算消耗时间,提高方案的安全性。安全性分析结果表明该算法能够抵抗当前主流的基于 LFSR 设计的流密码相关攻击,且类自同步流密码的设计能增强认证码的安全性。与 AES-CGM 和 AEGIS 的效率实验对比表明,在数据规模大的环境下,所提算法的效率高于 AES-CGM,与 AEGIS 的效率相当,具备一定的实用性。

关键词:祖冲之算法;流密码;认证加密;类自同步;关联数据
中图法分类号 TP309

Authenticated Encryption Scheme of Self-synchronous-like ZUC Algorithm

XU Rui,PENG Changgen and XU Dequan
College of Computer science and Technology,Guizhou University,Guiyang 550025,China
State Key Laboratory of Public Big Data,Ministry of Education,Guiyang 550025,China

Abstract Aiming at the security,efficiency and lightweight requirements of authentication encryption of ZUC algorithm,this paper proposes a kind of self-synchronous-like ZUC algorithm for associated data authentication encryption scheme ZUCAE. By improving the LFSR layer algorithm of ZUC stream cipher algorithm(ZUC-256),the scheme designs and implements a ZUC-SSL algorithm similar to self synchronous stream cipher,and uses this algorithm to make the ciphertext participate in the state update function for the generation of authentication code. This scheme encrypts the message through ZUC-256 algorithm,optimizes the initialization module,embeds the associated data into the initialization process,realizes the parallel generation of keystream and encryption,and authenticates the message before decryption,which reduces the calculation time and increases the security of the scheme. Security analysis results show that the algorithm can resist the current mainstream stream cipher related attacks based on LFSR design,and the design of self-synchronous-like stream cipher can enhance the security of authentication code. Compared with the efficiency experiments of AES-CGM and AEGIS,the results show that in the environment of large data scale,the efficiency is higher than that of AES-CGM,and is equivalent to AEGIS,so it has certain practicality.

Keywords ZUC,Stream cipher,Authentication encryption,Self-synchronizing like,Associated data

1 引言

认证加密算法是具备加密算法和生成消息鉴别码功能的算法,它可以在保护数据机密性的同时保证数据的完整性和数据源的认证。早期,国内外学者通过组合加密算法和消息鉴别码算法设计了 3 种认证加密算法:先加密后认证、先认证后加密、认证加密同时进行。这 3 种方式的优势在于具有通用性,适用于常见的加密算法和消息鉴别码,并且独立地将加密和消息鉴别码结合起来进行安全构建是相对灵活的,因此其逐渐成为一种既能提供机密性又能提供完整性和真实性的密码原语,但难以达到人们所需要的安全性[1]。因此,在 2000 年初,Bellare 等[2]分析了最初的 3 种认证加密算法的

到稿日期:2022-08-01 返修日期:2022-11-07
基金项目:国家自然科学基金重点项目(U1836205);贵州省科技计划重大专项项目(黔科合重大专项字[2018]3001);贵州省科技计划项目(黔科合平台人才[2020]5017,黔科合支撑[2018]2159)
This work was supported by the Key Program of the National Natural Science Foundation of China(U1836205),Guizhou Province Science and Technology Plan Project Major Special Project (Qian-Science-Contract-Major-Special-Project [2018]3001) and Guizhou Science and Technology Plan Project(Qian-Science-Contract-Platform-Talent[2020]5017,Qian-Science-Contract-Supporting [2018]2159).
通信作者:彭长根(peng_stud@163.com)

安全性和结构后，正式提出认证加密的概念。2002 年 Rogaway[3]提出了基于相关数据的认证加密算法，将部分需要路由的信息以明文方式传输并用于密钥产生。目前常用于设计的认证加密算法主要有两类：(1)是基于分组密码工作模式进行认证加密；(2)直接设计的认证加密算法，其中主要包括基于分组密码的认证加密算法、基于杂凑函数的认证加密算法和基于流密码的认证加密算法。传统密码学中，流密码通常首选同步方案，而不是自同步方案，这是因为它们产生了更高的潜在加密质量。尽管如此，自同步加密方案在正确实现时，具有将信息安全性与同步功能相结合的优势。2005 年，Daemen 等[4]提出了一种面向硬件的自同步流密码 Mosquito 并提交到 eStream 密码竞赛，后续也提供了多个版本以解决其选择密文攻击和易中断等问题。其后大部分设计的自同步流密码难以达到安全标准，直到将混沌理论和新型自动机等加入到密码算法中，之后开始大量设计自同步流密码，如数字混合消息嵌入自同步流密码[5]和自同步的非三角迭代结构安全传输算法 Stanislas[6]，以使自同步流密码的实现方式更加简单，提高其吞吐量。

为统一认证加密的标准，促进算法的兼容性，国际标准组织和 NIST 等机构收录了 OCB2.0，KeyWrap，CCM，EAX，GCM 等认证加密模式标准[7]。随着认证加密算法的性能和安全性等需求的日渐提升，2013 年，CAESAR 竞赛[8]征集到 57 种算法，经过 3 轮评估筛选出 15 种优胜算法，其中基于流密码的认证加密算法有 ACORN，MORUS，Raviyoyla，Sablier，TriviA-ck 和 Wheesht。它们均包含流密码初始化过程、关联数据处理过程、明文加密过程、后期白化和摘要生成过程这 4 个阶段，其中状态更新函数和数据块参与状态更新的方式是算法的核心部分。2018 年，为了将认证加密算法部署到资源受限的设备上，NIST 开始征集标准轻量级认证加密方案[9]，直到 2021 年，它们从第二轮的 32 种候选算法中选出 10 种优胜算法，作为标准的最终组合。认证加密算法经过长期大量的设计，安全性得到了提升，具有一定的可靠性，算法进行相应的优化后效率有所提高，基于流密码算法的认证加密算法也有了新的进展，尤其是近两年，使用各类算法进行认证加密的算法不断涌现，AEGIS[10]，ALE[11]，APE[12]等算法在运行速度和轻量级要求上均达到较高水平。但是针对我国首个国际标准流密码 ZUC 的认证加密算法及方案仍是难点。目前的主要方法是基于 ZUC 流密码算法已设计的消息鉴别码功能进行认证加密，但其单纯使用密钥流作为消息鉴别码；另一种是简单地利用 ZUC 流密码加密算法结合消息鉴别码进行认证加密。两种方法都难以达到安全性等要求。

因此本文首先给出一个基于关联数据的总体认证加密方案；然后根据自同步流密码和 ZUC 流密码算法的初始化过程，设计一种类自同步的 ZUC 流密码算法，用于标签生成；最后提出一种基于类自同步 ZUC 流密码的关联数据认证加密方案，并进行安全性分析和效率评估。

本文的主要贡献包括：

(1)设计了一种类自同步 ZUC 流密码算法，提高了认证码生成、认证的安全性。

(2)设计了实现基于国产流密码 ZUC 及其改进算法类自同步 ZUC 流密码的高效、安全的认证加密方案。

2 预备知识

2.1 自同步流密码

自同步流密码[13-14]与密文相关，即产生的密钥流除了与密钥相关以外，还与固定位数的密文位有关，是一种带有记忆变换的流密码。其模型如图 1 所示。

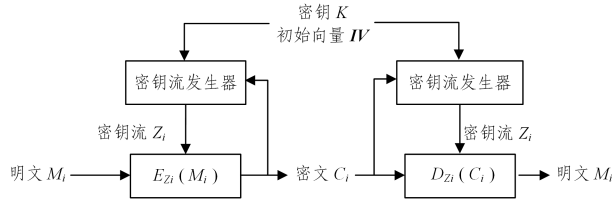

图 1 自同步流密码加解密模型

Fig. 1 Model of self-synchronous stream cipher

其加解密过程如下：

加密：$S_i=f(C_{i-t},C_{i-t+1},\cdots,C_{i-1})$

$Z_i=g(S_i,K)$

$C_i=E_{Z_i}(M_i)$

解密：$S_i=f(C_{i-t},C_{i-t+1},\cdots,C_{i-1})$

$Z_i=g(S_i,K)$

$M_i=D_{Z_i}(C_i)$

其中，S_i 是当前寄存器状态，f 是状态转换函数，g 是密钥流生成函数，Z_i 是密钥流，S_0 是寄存器初始状态，只由密钥 K 和初始向量 $\boldsymbol{IV}$ 决定。

自同步流密码具有有限的错误传播和自同步的特性：

(1) 有限的错误传播

自同步流密码的密钥流发生器具有固定数量的位存储时，若有一位错误的密文进行传输，则会影响后面固定数量位的解密。

(2)自同步

由有限的错误传播可知，当接收固定数量的正确明文符号时，双方的密钥会根据正确的明文位进行密钥自同步。

相对而言，自同步流密码的密文流参与密钥流的生成，使密钥流的分析复杂化，导致自同步流密码的设计更加困难，也较难从理论上进行详尽的分析。

2.2 ZUC 流密码

ZUC-256 算法[15]是 ZUC-128 算法[16]的 256bit 密钥升级版本，其设计目标是在 5G 的应用环境下提供 256bit 的安全性。ZUC 族流密码主要包含 3 层算法结构，分别为线性反馈移位寄存器层(LFSR)、比特重组层(BR)和非线性函数层(FSM)。

上层 LFSR 中的设计首次采用素域 $GF(2^{31}-1)$的 m 序列，由 16 个 31 比特寄存器单元$(s_{15},s_{14},\cdots,s_1,s_0)$构成，其过程主要为更新移位寄存器内部状态，其中初始化阶段中移位寄存器更新过程如下：

$LFSRWithInitializationMode(u)\{$

$v=(2^{15}\cdot s_{15}+2^{17}\cdot s_{13}+2^{21}\cdot s_{10}+2^{20}\cdot s_4+(1+2^8)\cdot s_0) \bmod (2^{31}-1)$

若 $v=0$，则 $v=2^{31}-1$；

$s_{16}=(v+u) \bmod (2^{31}-1)$

若 $s_{16}=0$，则 $s_{16}=2^{31}-1$；

$(s_{16},s_{15},\cdots,s_2,s_1)\rightarrow(s_{15},s_{14},\cdots,s_1,s_0);\}$

密钥流生成阶段中的移位寄存器更新过程如下：

$LFSRWithworkMode()\{$

$s_{16}=(2^{15}\cdot s_{15}+2^{17}\cdot s_{13}+2^{21}\cdot s_{10}+2^{20}\cdot s_4+(1+2^8)\cdot s_0)\bmod(2^{31}-1)$

若 $s_{16}=0$，则 $s_{16}=2^{31}-1$；

$(s_{16},s_{15},\cdots,s_2,s_1)\rightarrow(s_{15},s_{14},\cdots,s_1,s_0);\}$

其次是一个比特重组层(BR)，比特重组的过程如下：

$Bitreorganization()\{$

$X_0=s_{15H}\parallel s_{14L};$

$X_1=s_{11H}\parallel s_{9L};$

$X_2=s_{7H}\parallel s_{5L};$

$X_3=s_{2H}\parallel s_{0L};\}$

最后是 FSM 层，非线性函数如下：

$F(X_0,X_1,X_2)\{$

$W=(X_0\oplus R_1)(R_2;$

$W_1=R_1(X_1;$

$W_2=R_2\oplus X_2;$

$R_1=S(L_1(W_{1L}\parallel W_{2H}));$

$R_2=S(L_2(W_{2L}\parallel W_{1H}));\}$

其中 $S=(S_0,S_1,S_0,S_1)$为 4 个并置的 S 盒；L_1 与 L_2 采用两个 MDS 矩阵。ZUC 流密码通过 3 层的变换更新得到密钥流，与明文异或加密得到密文，实现轻量化的安全加密过程。

3　类自同步 ZUC 关联数据认证加密方案

3.1　符号说明

符号说明如表 1 所列。

表 1　符号说明

Table 1　Explanation of symbols

符号	说明
K	用户密钥，$K\in\{0,1\}^{256}$
$\boldsymbol{N}$	Nonce，$N\in\{0,1\}^{256}$
$\boldsymbol{IV}$	初始向量
AD	关联数据
$adlen$	关联数据长度
s	LFSR 的内部状态
s_i	第 i 位 LFSR 的内部状态
M	明文
M_i	明文的第 i 块
C	密文
C_i	密文的第 i 块
H	密文的哈希值
$msglen$	明文/密文长度
KS	密钥流
T	标签，$T\in\{0,1\}^{256}$

3.2　方案描述

基于 ZUC 的认证加密算法的输入包含 ZUC 密码算法的用户密钥 K、初始化向量 $\boldsymbol{N}$(即 Nonce)、关联数据 AD、明文消息 M；输出密文消息 C 和标签 T。本方案首先根据 ZUC-256 流密码的特性，处理 AD 和 $\boldsymbol{N}$，截取处理结果的前 184 bit 作为 $\boldsymbol{IV}$；再将 $\boldsymbol{IV}$ 和 K 作为 ZUC 流密码的输入，生成与长度为 $msglen$ 的密钥流 KS，加密获得密文 C；最后将 C 作为 hash 函数的输入，得到 256 bit 的 hash 值 H，将 H 输入类自同步的 ZUC 密码算法(ZUC-SSL)，生成标签 T。该方案定义为：

$ZUCAE(AD,\boldsymbol{N},K,M)=(C;M\oplus KS;T;ZUC\text{-}SSL(Hash(C)))$

3.3　类自同步流密码 ZUC 算法(ZUC-SSL)

本方案结合自同步流密码和 ZUC 算法初始化过程的思想，将密文加入到线性反馈移位寄存器中，参与密钥流的生成过程，提出了一种密文相关的类自同步 ZUC 流密码。其过程为将抽头计算相加得到结果后，再将密文 C 右移一位，使密文的高 31 位 C_{31h} 与抽头计算结果相加后进行 $\bmod(2^{32}-1)$ 计算，得到线性反馈移位寄存器的下一位，更新至下一状态中，其他层算法与 ZUC 算法相同。由于其连续性错误和密钥流难以分析等特性，密文复杂且不可预测，因此将该算法应用于认证算法中，实现认证加密的认证功能，能保证其认证过程中的安全性。算法结构如图 2 所示。

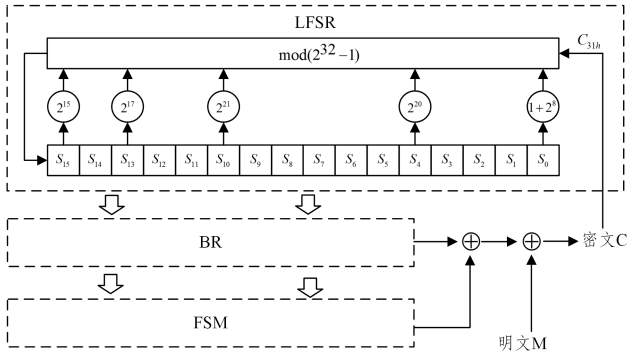

图 2　ZUC-SSL 算法结构图

Fig. 2　Structure diagram of ZUC-SSL

类自同步 ZUC 算法的具体步骤如算法 1 所示。

算法 1　类自同步 ZUC 算法

输入：K，**IV**，M

输出：C

1. ZUC_Load(K，**IV**)；/ * 初始化 ZUC * /
2. for i=1 to 32
3.　　Bitreorganization( )；
4.　　W=F(X0，X1，X2)；
5.　　LFSRWithInitializationMode(W≫1)；
6. endfor
7. Bitreorganization( )；
8. W=F(X0，X1，X2)；
9. LFSRWithworkMode()；
10. for i=1 to msglen/32
11.　　Bitreorganization()；
12.　　$KS_i=F(X_0,X_1,X_2)\oplus X_3$；
13.　　$C_i=KS_i\oplus M_i$；
14.　　LFSRWithSSLMode($C_i\gg 1$)；
15. endfor
16. $C\leftarrow(C_1,C_2,\cdots,C_{(msglen/32)-1},C_{msglen/32})$；

其中 $LFSRWithSSLMode(C_{31h})$ 状态函数的主要构成如下：

$LFSRWithSSLMode(C_{31h})\{$

$v=(2^{15}\cdot s_{15}+2^{17}\cdot s_{13}+2^{21}\cdot s_{10}+2^{20}\cdot s_4+(1+2^8)\cdot s_0)\bmod(2^{31}-1);$

if $v=0$：$v=2^{31}-1$；

$s_{16}=(v+C_{31h})\bmod(2^{31}-1)$；

if $s_{16}=0$：$s_{16}=2^{31}-1$；

$(s_{15},s_{14},\cdots,s_1,s_0)\leftarrow(s_{16},s_{15},\cdots,s_2,s_1);\}$

3.4 方案具体流程

(1)初始化过程:主要包括关联数据处理和 ZUC 流密码初始化过程。首先得到 AD 的 256 位 hash 值,将 hash 值与 $\boldsymbol{N}$ 异或得到 ZUC 流密码的初始向量 $\boldsymbol{IV}$,将 $\boldsymbol{IV}$ 和 K 注入 ZUC 流密码算法,进行 33 轮的 ZUC 初始化,并将初始化结果状态 $s_i(i=0,1,2,\cdots,15)$ 赋值到标签过程状态 s_i^* $(i=0,1,2,\cdots,15)$ 用于标签生成。ZUCAE 初始化过程如算法 2 所示。

算法 2 ZUCAE 初始化

输入:AD,**N**,K
输出:s,s*
1. **IV**=(Hash(AD)⊕**N**);
/* 利用关联数据生成初始 **IV** */
2. ZUC_Load(K,IV);/* 初始化 ZUC */
3. for i=1 to 32
4. Bitreorganization( );
5. W=F(X_0,X_1,X_2);
6. LFSRWithInitializationMode(W≫1);
7. endfor
8. Bitreorganization();
9. W=F(X_0,X_1,X_2);
10. LFSRWithworkMode();
11. s←s* /* 生成标签过程的状态保存 */

(2)明文加密过程:ZUC 初始化完成后,在状态 $s_i(i=0,1,2,\cdots,15)$ 的基础上进行密钥流 KS 生成,其中 ZUC 流密码每一节拍产生一个 32 比特字作为输出密钥流,因此需要生成 $l/32$ 个密钥流。密钥流生成与明文加密同步进行,利用 KS 和明文 M 异或得到密文 C,直至所有明文加密完成,具体过程如算法 3 所示。

算法 3 ZUCAE 加密算法

输入:M,s
输出:C
1. for i=1 to msglen/32
2. Bitreorganization();
3. KS_i=F(X_0,X_1,X_2)⊕X_3;
4. C_i=KS_i⊕M_i;
5. LFSRWithworkMode();
6. endfor
7. C←(C_1,C_2,…,$C_{(msglen/32)-1}$,$C_{msglen/32}$).

(3)标签生成过程:利用 hash 函数生成 C 的 hash 值 H,将初始化过程中备份的初始化结果装载进 ZUC-SSL 算法中;然后将 H 作为类自同步的 ZUC 流密码算法 ZUC-SSL 的输入,逐位生成 256 位密钥流,作为 T。具体过程如算法 4 所示。

算法 4 ZUCAE 标签生成算法

输入:C,s*
输出:T
1. s←s*
2. H=Hash(C);
3. for i=1 to 8
4. Bitreorganization();
5. KS_i=F(X_0,X_1,X_2)⊕X_3;
6. T_i=KS_i⊕H_i;
7. LFSRWithSSLMode(T_i);
8. endfor
9. T←(T_1,T_2,…,T_8).

(4)解密和认证过程:ZUC-SSL 中的 LFSR 状态已预先初始化完成,因此无须先解密再认证。认证成功将返回明文,否则将不再解密密文,仅输出认证失败的结果。

4 安全性分析

在进行安全分析之前,需要满足以下要求:

(1)密钥的生成是安全的。

(2)每个密钥和 $\boldsymbol{IV}$ 对只能使用一次。

(3)验证失败后,将不继续解密密文,仅输出验证失败的结果。

如果满足上述要求,我们有以下安全声明:

(1)若伪造攻击的成功率为 2^{-t},其中 t 是标签 T 的长度。当伪造攻击重复 n 次时,伪造攻击的成功率约为 $n\times 2^{-t}$。

(2)若伪造攻击未成功,则恢复状态和密钥的速度将比穷举密钥搜索更慢。

本章将分析初始化过程、加密过程、消息认证过程中的安全性。其中主要分析加密和消息认证过程的安全性。

4.1 初始化过程的安全性

在初始化过程中,初始向量的差分都会传播到密钥流中,通过分析初始向量和密钥流之间的关系,可能会产生差分攻击。在 ZUCAE 中,通过 ZUC 自身的 33 轮初始化,如果有一个 $\boldsymbol{IV}$ 差分,那么这个差分会经过 33 次以上的状态更新,使得线性移位寄存器中的状态无法被分析,即预计对于初始化的差分攻击将会比穷尽密钥搜索花费的代价更高。此外,通过 33 轮的初始化,可以抵御包括差分攻击在内的线性攻击和立方体攻击等。

4.2 加密过程的安全性

ZUCAE 算法的加密过程由关联数据处理函数和 ZUC 的加密算法构成。算法的关联数据处理函数包含了 hash 函数、hash 值与 Nonce 处理,可以确定对 ZUC 的同一对密钥 K 和初始向量 $\boldsymbol{IV}$ 不会重复使用。否则,已知明文攻击或选择明文攻击都可以轻易地恢复状态值,分析 $\boldsymbol{IV}$ 和密钥流的关系。

针对基于线性反馈移位寄存器的流密码算法进行攻击主要存在 4 种分析,分别是弱密钥分析、线性区分分析、代数分析和猜测决定分析。

对于 ZUC 算法,2012 年 Wu 等[17]修改了早期设计的非线性函数 F 通过直接异或的方式参与 LFSR 的更新,重新保证初始化过程中更新函数的单向性,确保初始化过程是复杂的非线性迭代,即可以将其看成一个随机置换。其中弱密钥的概率大概为 $2^{64}\times 2^{-304}=2^{-240}$,因此加密过程中不太可能存在弱密钥分析。2016 年,Feng[18]对 ZUC 进行了简要的代数分析,指出若攻击者截获 18 个子密钥,则至少涉及 12 010 个线性独立的二次代数方程,利用代数分析的方法求解几乎不可能。

上述两种方法都难以对 ZUC 算法构成威胁,因此目前针对 ZUC 算法安全性分析最多的是线性区分分析和猜测决定分析。2016 年,Tang 等[19]对 ZUC-128 算法中两轮非线性函数 F 最优化线性逼近,得到线性逼近方程式,计算线性逼近的偏差,该攻击需要 $O(2^{131}$ bit)密钥流。2013 年,Guan 等[20]将 ZUC-128 算法中基于 32 比特字的非线性函数转化为基于 16 比特半字的非线性函数,提出了基于 16 比特半字的猜测决定攻击,将计算复杂度降低至 $O(2^{392})$。2019 年,Wang 等[21]将 ZUC-256 中的状态转换运算变换为半位的运算,将

计算复杂度降低至$O(2^{358})$，需截取密钥流的数据量为 7 个 32 比特密钥字。虽然这两个方案的复杂度都低于最初方案[22]，但从表 2 中现有猜测攻击方案和本文线性区分攻击方案的计算复杂度对比可知，它们对 ZUC-256 的攻击效率仍低于穷举密钥搜索算法。

表 2　已有的针对 ZUC 的分析结果比较

Table 2　Comparison of existing analysis results for ZUC

	攻击方法	攻击复杂度
Ding	猜测决定攻击	$O(2^{403})$
Guan 等	猜测决定攻击	$O(2^{392})$
Wang 等	猜测决定攻击	$O(2^{358})$
Tang 等	线性区分分析	$O(2^{131})$

4.3　消息认证过程的安全性

ZUCAE 算法的消息认证过程由 hash 函数和类自同步的 ZUC 算法构成。这里主要考虑类自同步的 ZUC 算法 ZUC-SSL 对消息认证过程的保护作用。

在 ZUC-SSL 中密文反馈到 LFSR 层中，ZUC 流密码的 LFSR 层共有 16 个存储单元 s_i，当解密过程中反馈密文被截获或篡改时，错误密文的 C_{31h} 将直接影响 LFSR，得到错误的更新状态 s_{15}，导致后续存储单元的 s_i 状态值陆续出错。可见，如果伪造攻击可以对密钥 K 和初始向量 $\boldsymbol{IV}$ 对进行重复运算，攻击者在标签 T 的认证过程中引入差分值，由于 ZUC-SSL 输出的认证码为 256bit，则获取解密明文至少需要进行 2^{256} 次伪造攻击，同时 ZUC-SSL 也进行了 33 轮初始化来将差分值扩散，保证了伪造攻击的不可行。

针对猜测决定攻击，我们在文献[21]的基础上设计并计算猜测决定攻击的复杂度，其中进行猜测的内部状态共 365bit 以及其中一位密文反馈位 C_{31h} 的 31 bit，此时的猜测量为 2^{396}。通过验证所猜测的比特位是否正确，判定验证式是否一定相等，发现设计的算法中共有 7 个过程可实现猜测判定正确与否。因此该攻击对 ZUC-SSL 的计算复杂度为 2^{389}，说明其具有较强的抵抗猜测决定攻击的能力。

我们针对时间存储数据折中攻击进行了分析。其中，设在线阶段的时间复杂度为 T，存储复杂度为 M，攻击所需的数据量为 D；离线攻击阶段的时间复杂度为 P，流密码算法中的全部内部状态为 N。在 ZUC-SSL 算法中内部状态为 $16\times31+2\times32+31=591$ bit，因此 $N=2^{591}$。我们主要考虑了两种常用的时间存储折中攻击方法。

(1)BG 方法。折中曲线为 $MD\geqslant N$，$TM=N$，$P=M$，$T=D$，因为 $N=2^{591}$，可知 M 和 D 两者至少有一个的复杂度高于 2^{296}，因超出了穷举的复杂度，故攻击不可行。

(2)BS 方法。折中曲线为 $TM^2D^2\geqslant N^2$，$T\geqslant D^2$，$N=PD$，因为 $N=2^{591}$，可计算 $M=D=N^{1/3}=2^{197}$，$T=D^2=2^{394}$，因此该攻击的离线攻击事件复杂度、存储复杂度和在线攻击复杂度很高，攻击同样不可行。

因此，类自同步的 ZUC 流密码算法(ZUC-SSL)增强了 LFSR 层的复杂性，提升了密钥流输出的复杂度以及对密钥流分析的复杂程度，能有效抵抗伪造攻击、猜测决定攻击和时间存储数据攻击等，增加了消息认证的安全性。

5　实验结果与分析

为检测 ZUCAE 算法的效率，本文在处理器为 Intel core i7-10750H 的计算机上对其进行了实现，编译环境为 Visual Studio 2019，语言采用 C++。分别对 AES-GCM、AEGIS[10] 以及本方案进行对比。其中本方案中 hash 算法采用 SM3，测评主要针对加密与标签生成模块进行，在相同消息规模下，进行多组测评并取平均值。将所有数据分为文本数据和图片数据，表 3 和图 3 给出了 3 种算法关于文本数据的实验结果对比；表 4 和图 4 给出了 3 种算法关于图片数据的实验结果对比。

表 3　算法运行效率对比(文本数据)

Table 3　Comparison of algorithm operation efficiency(in text data)

消息长度	AEGIS/ms	AES-CGM/ms	本文方案/ms
128 bit	0.091	0.087	0.163
512 bit	0.170	0.160	0.250
1 024 bit	0.280	0.330	0.340
1 kB	0.840	1.250	1.030
5 kB	3.510	4.230	3.950

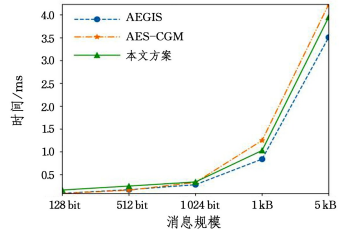

图 3　算法运行效率对比图(文本数据)

Fig. 3　Comparison diagram of algorithm operation efficiency (in text data)

从表 3 和图 3 可以看出，在文本数据中消息规模不大的情况下 ZUCAE 算法效率较低，当数据规模达到 1 024 bit 时，3 种算法效率接近。当数据规模达到 1 kB 时，流密码加密的优势得到体现，其效率开始高于基于 AES 的计数器模式下的认证加密方式(AES-GCM)。由表 4 和图 4 可以看出，在图像数据下 ZUCAE 算法效率高于 AES-GCM，与 AEGIS 算法效率相当。增加了类自同步的 ZUC 认证功能之后，生成标签的时间消耗并不多，主要消耗时间为明文加密过程。这说明消息规模越大，ZUCAE 算法的认证加密效率越高，并且逐渐接近 CEASER 竞赛的优选算法 AEGIS。综合考虑，ZUCAE 算法无须解密即可进行认证，其效率较高。

表 4　算法运行效率对比(图片数据)

Table 4　Comparison of algorithm operation efficiency(in picture data)

消息长度/kB	AEGIS/ms	AES-CGM/ms	本文方案/ms
1	0.97	1.36	1.16
2	1.74	2.17	1.97
5	3.87	4.13	4.04
10	6.18	7.09	6.45

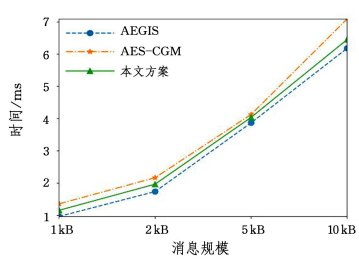

图 4　算法运行效率对比图(图片数据)

Fig. 4　Comparison diagram of algorithm operation efficiency (in picture data)

结束语 本文在国密 ZUC 流密码算法的基础上，结合 ZUC 流密码算法初始化和自同步流密码的思想，设计类自同步 ZUC 算法用于认证码生成。提出了一种关联数据认证加密方案，增强了初始化阶段对于 ***IV*** 的差分攻击等的抵抗功能；实现了加密阶段的密钥流生成和加密并行，降低了加密过程的时间消耗；利用类自同步 ZUC 算法增加了详细认证的安全性；将认证过程设计到解密之前，进一步提高了密文安全性。通过对 ZUCAE 的具体操作过程的描述，对初始化、加密过程以及认证阶段进行了安全性分析，表明该算法能够抵抗大部分攻击，能够满足算法的安全性需要。最后，对 ZUCAE 的加密过程进行了效率对比，发现算法的加密效率与其他认证加密算法效率相当，但其无须解密即可认证，可降低其认证过程的时间消耗，同时增加其安全性。因此，在下一步研究过程中，将针对 ZUCAE 算法继续进行改进，通过优化部分算法，提高 ZUCAE 算法的安全性和效率等。

参 考 文 献

[1] BELLARE M, KOHNO T, NAMPREMPRE C. Authenticate-dencryption in SSH: Provably fixing the SSH binary packet protocol[C]// Proceedings of the 9th ACM Conference on Computer and Communications Security. 2002: 1-11.

[2] BELLARE M, NAMPREMPRE C. Authenticated encryption: Relations among notions and analysis of thegeneric composition paradigm[J]. Journal of Cryptology, 2008, 21(4): 469-491.

[3] ROGAWAY P. Authenticated-encryption with associated-data [C]// Proceedings of the 9th ACM Conference on Computer and Communications Security. 2002: 98-107.

[4] DAEMEN J, KITSOS P. The self-synchronizing stream cipher moustique[M]//New Stream Cipher Designs. Berlin: Springer, 2008: 210-223.

[5] TANOUGAST C, WEBER S, MILLERIOUX G, et al. An Fpga implementation of the HME self-synchronizing stream cipher for Enhanced security and performance[C]// Second NASA/ESA Conference on Adaptive Hardware and Systems (AHS 2007). IEEE, 2007: 110-118.

[6] FRQNCQ J, BESSON L, HUYNH P, et al. Non-triangular self-synchronizing stream ciphers[J]. IEEE Transactions on Computers, 2020, 71(1): 134-145.

[7] WU W L. Research advances on authenticated encryption algorithms[J]. Journal of Cryptologic Research, 2018, 5(1): 70-82.

[8] ZHANG F, LIANG Z Y, YANG B L, et al. Survey of design and security evaluation of authenticated encryption algorithms in the CAESAR competition[J]. Frontiers of Information Technology & Electronic Engineering, 2019, 19(12): 1475-1499.

[9] TURAN M S, MCKAY K, CHANG D, et al. Status report on the second round of the NIST lightweight cryptography standardization process [R]. National Institute of Stan-dards and Technology Internal Report, 2021.

[10] WU H, PRENEEL B. AEGIS: A fast authenticated encryption algorithm[C]// International Conference on Selected Areas in Cryptography. 2013: 185-201.

[11] BOGDANOV A, MENDEL F, REGAZZONI F, et al. ALE: AES-based lightweight authenticated encryption[C]// International Workshop on Fast Software Encryption. 2013: 447-466.

[12] ANDREEVA E, BILGIN B, BOGDANOV A, et al. APE: authenticated permutation-based encryption for lightweight cryptography[C]// International Workshop on Fast Software Encryption. 2014: 168-186.

[13] LIU J Y. Applied cryptography[M]. Beijing: Tsinghua University Press, 2008: 165-171.

[14] UO J B, ZHANG J. Current situation and development of stream cipher[J]. Journal of Terahertz Science and Electronic Information Technology, 2006, 4(1): 75-80.

[15] Design Team. ZUC-256 Stream Cipher [J]. Journal of Cryptologic Research, 2018, 5(2): 167-179.

[16] FENG X T. ZUC Algorithm: 3GPP LTE International Encryption Standard[J]. Information Security and Communications Privacy, 2011, 9(12): 45-46.

[17] WU H, TAO H, NGUYEN P H, et al. Differential Attacks against Stream Cipher ZUC[C]// International Conference on the Theory & Application of Cryptology & Information Security. 2012.

[18] FENG X T. ZUC stream cipher algorithm[J]. Journal of Information Security Research, 2016, 2(11): 1028-1041.

[19] TANG Y L, HAN D, YAN X X, et al. Linear distinguishing attack analysis on ZUC stream cipher[J]. Journal of Nanjing University of Science and Technology, 2016, 40(4): 450-454.

[20] GUAN J, DING L, LIU S K. Guess and determine attack on SNOW3G and ZUC[J]. Journal of Software, 2013, 24(6): 1324-1333.

[21] WANG Z Y, MAO M, ZHANG Y S. Guess and determine attack on ZUC-256 stream cipher[J]. Journal of Computer Applications, 2019, 39(S1): 105-108.

[22] DING L, LIU S K, ZHANG Z Y, et al. Guess and determine attack on ZUC based on solvingnonlinear equations[C]// Proceedings of the 1st International Workshop on ZUC Algorithm. 2010.

XU Rui, born in 1996, postgraduate. His main research interests include cryptography and information security.

PENG Changgen, born in 1963, Ph. D, professor, Ph.D supervisor, is a professional member of China Computer Federation. His main research interests include cryptography, information security, and privacy protection of big data.

（责任编辑：何杨）