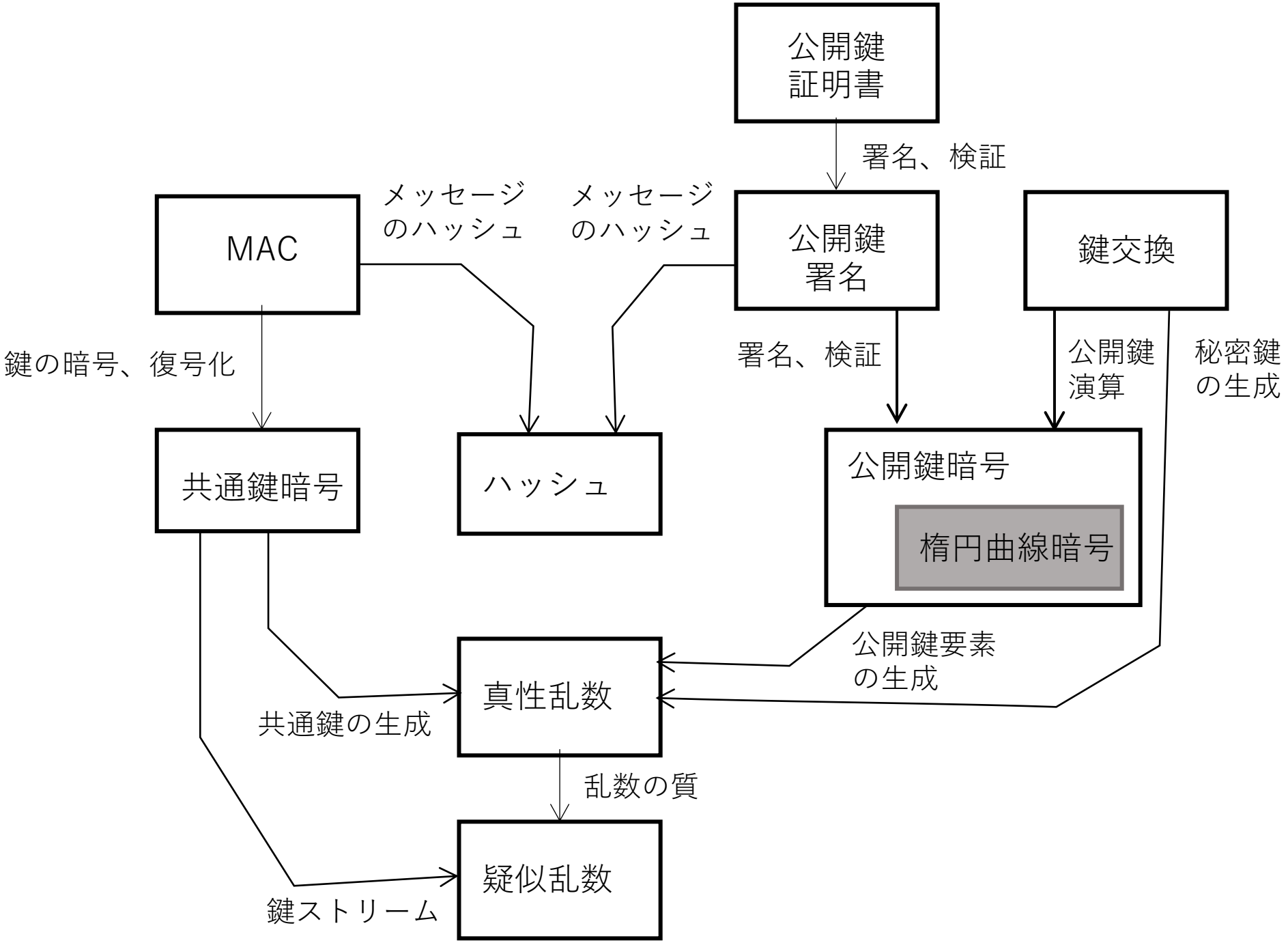
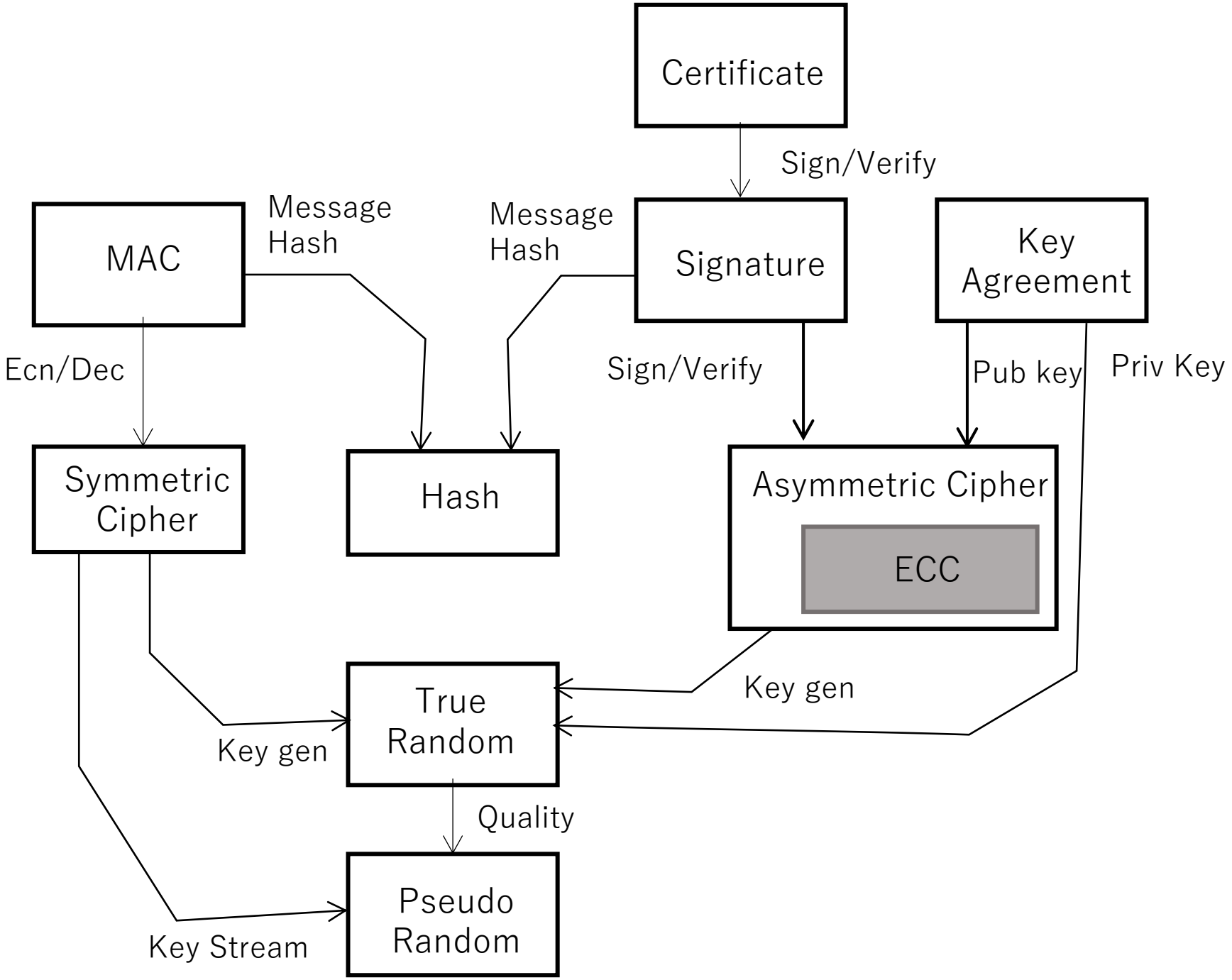
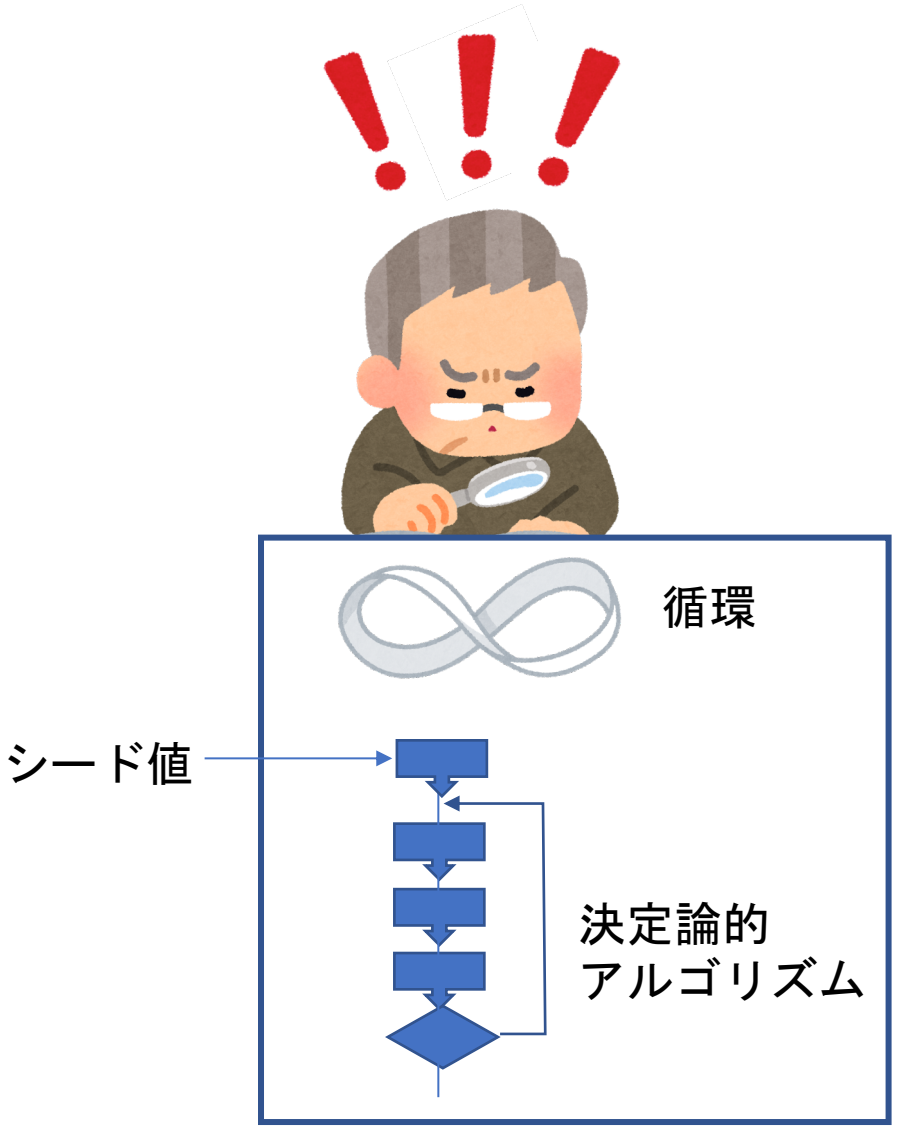

公開鍵
証明書
署名、検証
MAC
メッセージ
のハッシュ
メッセージ
のハッシュ
公開鍵
署名
鍵交換
鍵の暗号、復号化
署名、検証
公開鍵
演算
秘密鍵
の生成
共通鍵暗号
ハッシュ
公開鍵暗号
楕円曲線暗号
公開鍵要素
の生成
真性乱数
共通鍵の生成
乱数の質
疑似乱数
鍵ストリーム


Certificate
Sign/Verify
MAC
Message
Hash
Message
Hash
Signature
Key
Agreement
Ecn/Dec
Sign/Verify
Pub key
Priv Key
Symmetric
Cipher
Hash
Asymmetric Cipher
ECC
Key gen
True
Random
Key gen
Quality
Pseudo
Random
Key Stream

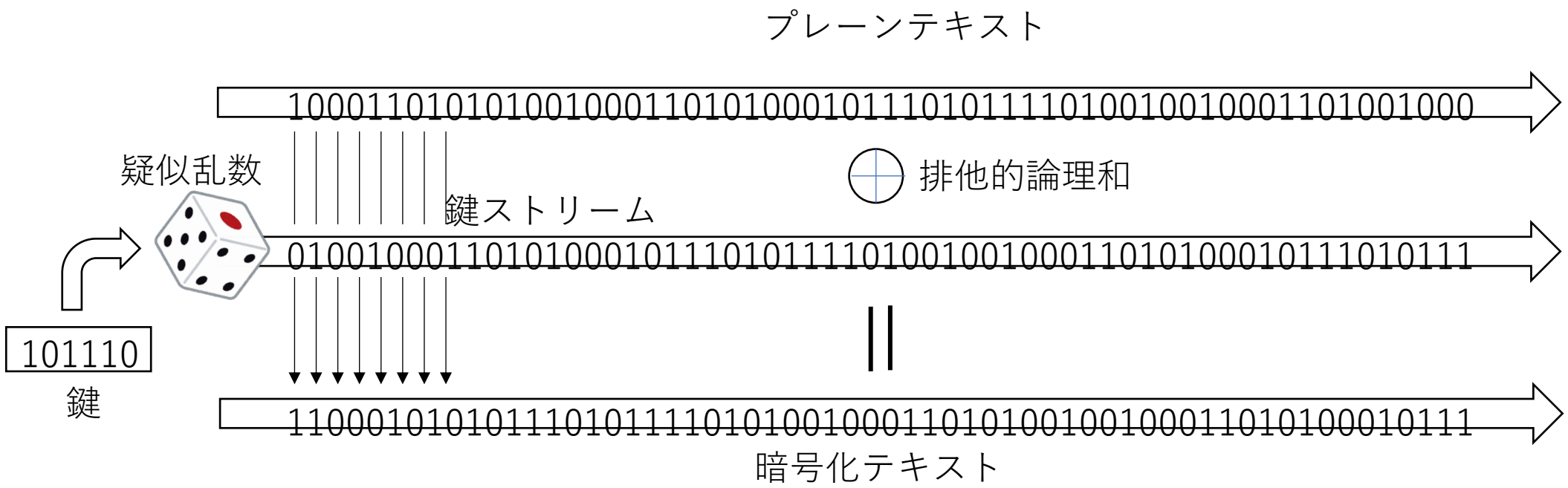
真性乱数



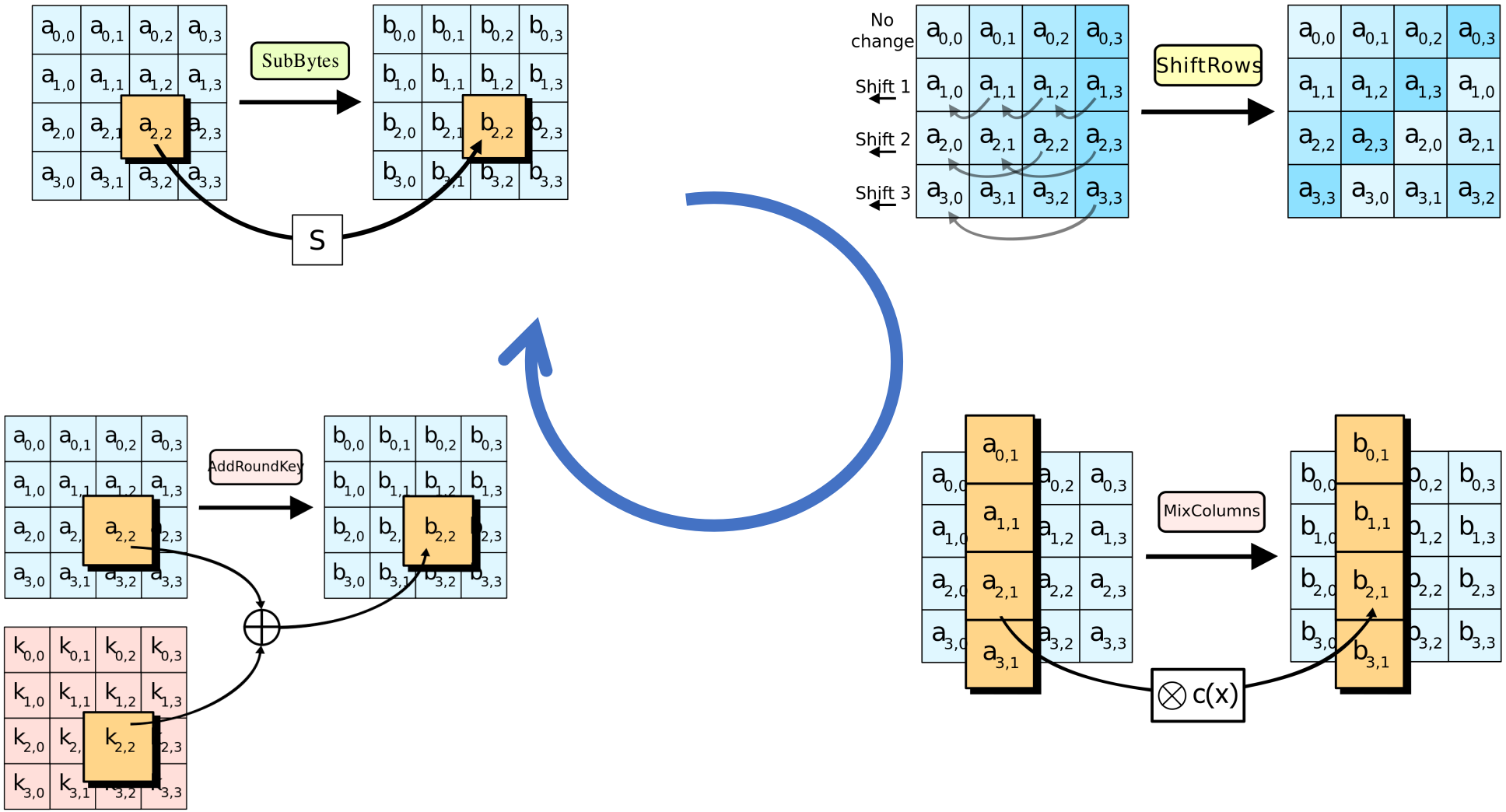
擬似乱数

アルゴリズムとバリエーション		出力長 (bits)	内部状態長 (bits)	ブロック長 (bits)	最大メッセージ長 (bits)	ラウンド数	ビット演算	セキュリティ強度 (bits)
MD5		128	128 (4 × 32)	512	$2^{64} - 1$	64	And, Xor, Rot, Add (mod 2^{32}), Or	<64（強衝突）
SHA-0		160	160 (5 × 32)	512	$2^{64} - 1$	80	And, Xor, Rot, Add (mod 2^{32}), Or	<80（強衝突）
SHA-1		160	160 (5 × 32)	512	$2^{64} - 1$	80		<63 (衝突発見[6])
SHA-2	*SHA-224*	224	256 (8 × 32)	512	$2^{64} - 1$	64	And, Xor, Rot, Add (mod 2^{32}), Or, Shr	112
	SHA-256	256						128
	SHA-384	384	512 (8 × 64)	1024	$2^{128} - 1$	80	And, Xor, Rot, Add (mod 2^{64}), Or, Shr	192
	SHA-512	512						256
	SHA-512/224	224						112
	SHA-512/256	256						128
SHA-3	*SHA3-224*	224	1600 (5 × 5 × 64)	1152	制限なし[7]	24[8]	And, Xor, Rot, Not	112
	SHA3-256	256		1088				128
	SHA3-384	384		832				192
	SHA3-512	512		576				256
	SHAKE128	d（可変長）		1344				d/2と128のいずれか小さい方
	SHAKE256	d（可変長）		1088				d/2と256のいずれか小さい方

ストリーム暗号


プレーンテキスト
1000110101010010001101010001011101011111010010010001101001000
疑似乱数
排他的論理和
鍵ストリーム
0100100011010100010111010111110100100100011010100010111010111
101110
鍵
||
1100010101011101011111010100100011010100100100011010100010111
暗号化テキスト

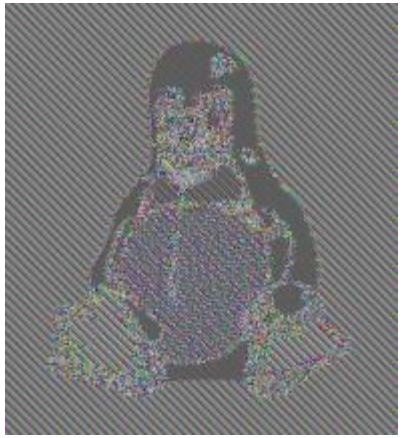
AES


SubBytes
S
No change
Shift 1
Shift 2
Shift 3
ShiftRows
AddRoundKey
MixColumns
c(x)

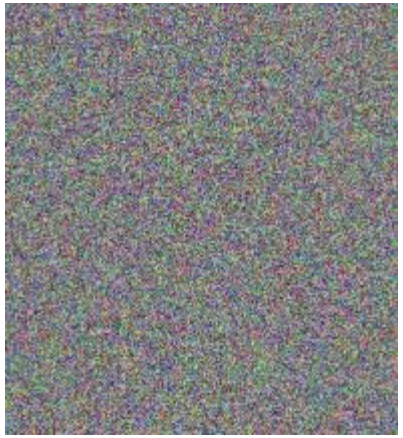
利用モード

Electronic Codebook (ECB)
Cipher Block Chaining (CBC)
Propagating Cipher Block Chaining (PCBC)
Cipher Feedback (CFB)
Output Feedback (OFB)
Counter (CTR)

元画像



ECBモードでの
暗号化



ECBモード以外での暗号化

右の画像は、CBC、CTRなどECBモード以外での暗号化における結果の例である。ランダムなノイズのように見えることが安全に暗号化されていることを必ずしも意味しないことには注意する必要がある。

Wikipedia：暗号利用モード

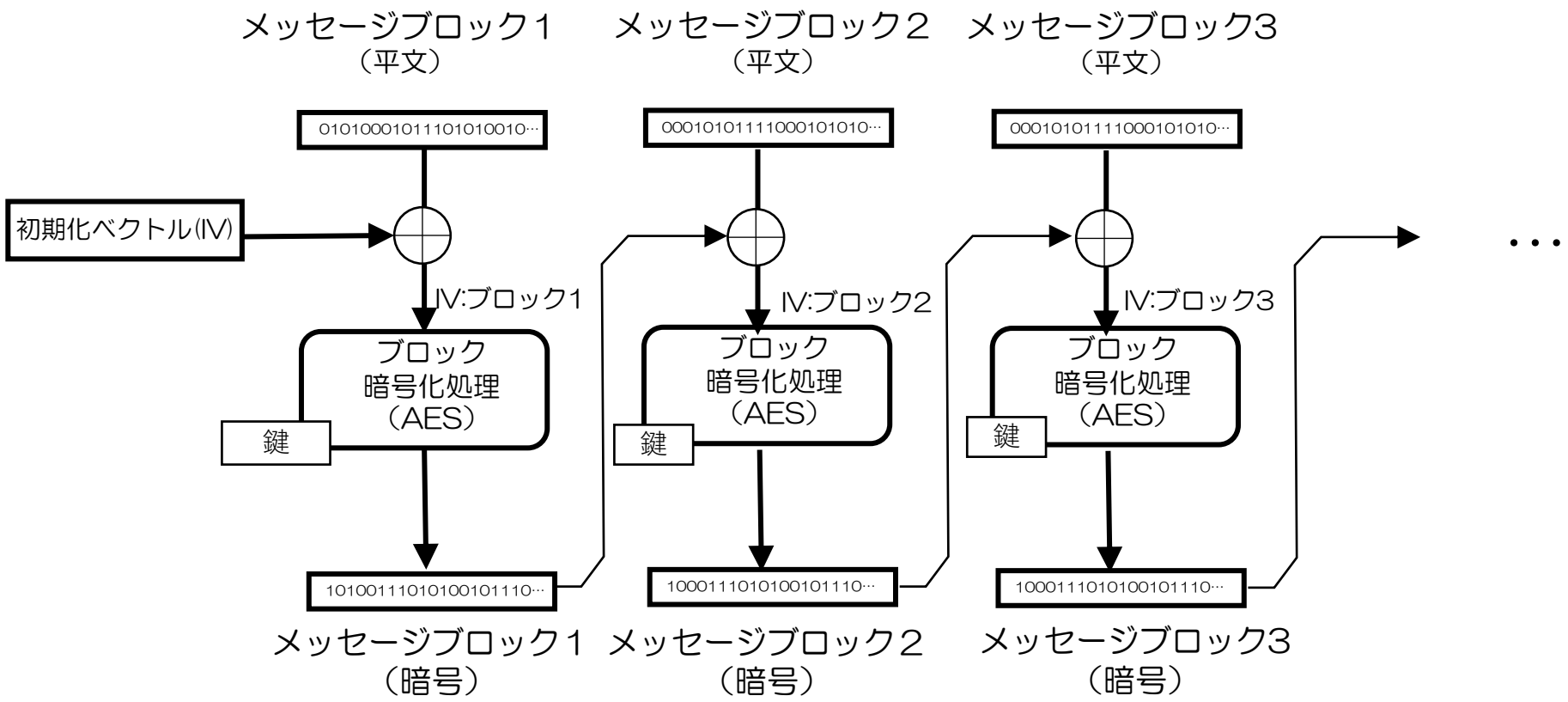


図3-4-2: 暗号ブロック・チェーン(CBC)

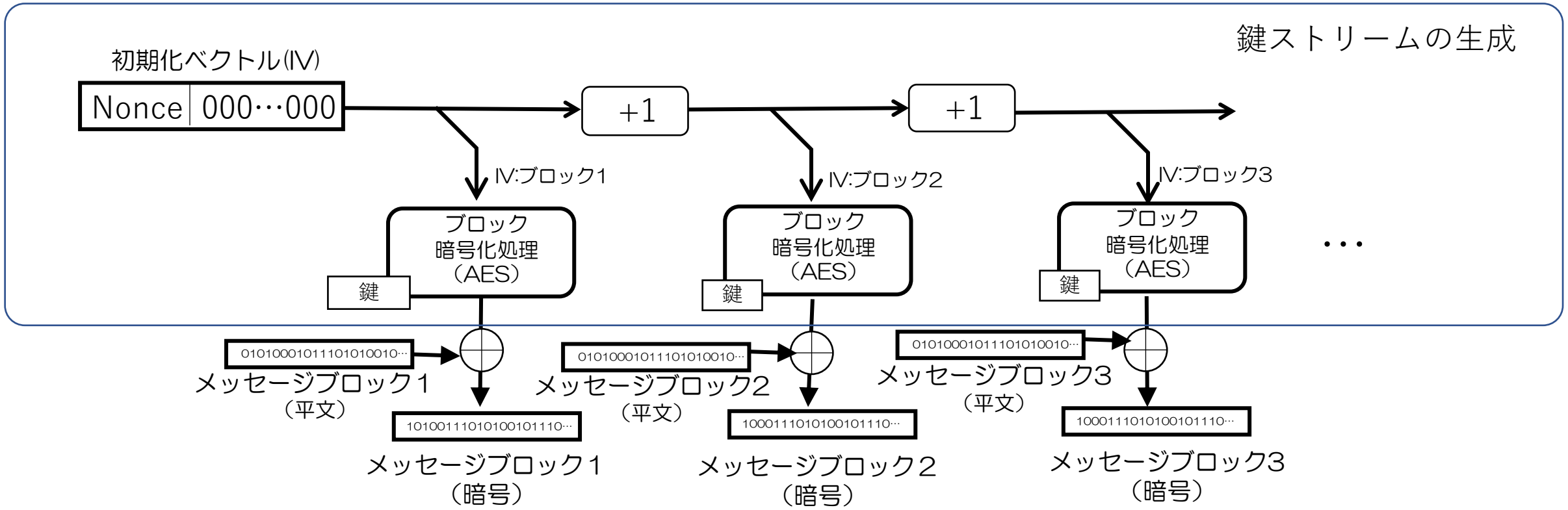


図3-4-3: カウンターモード(CTR)

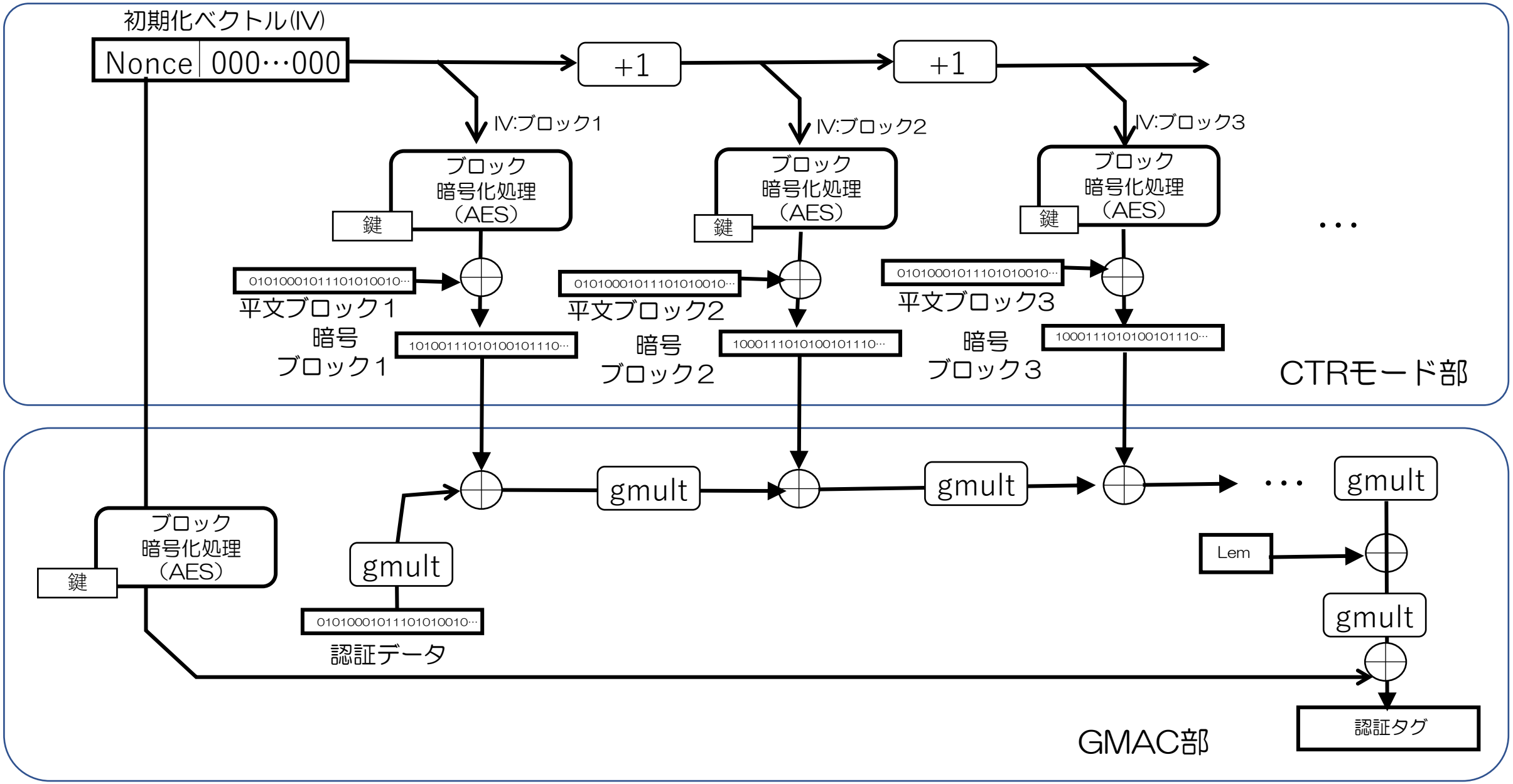
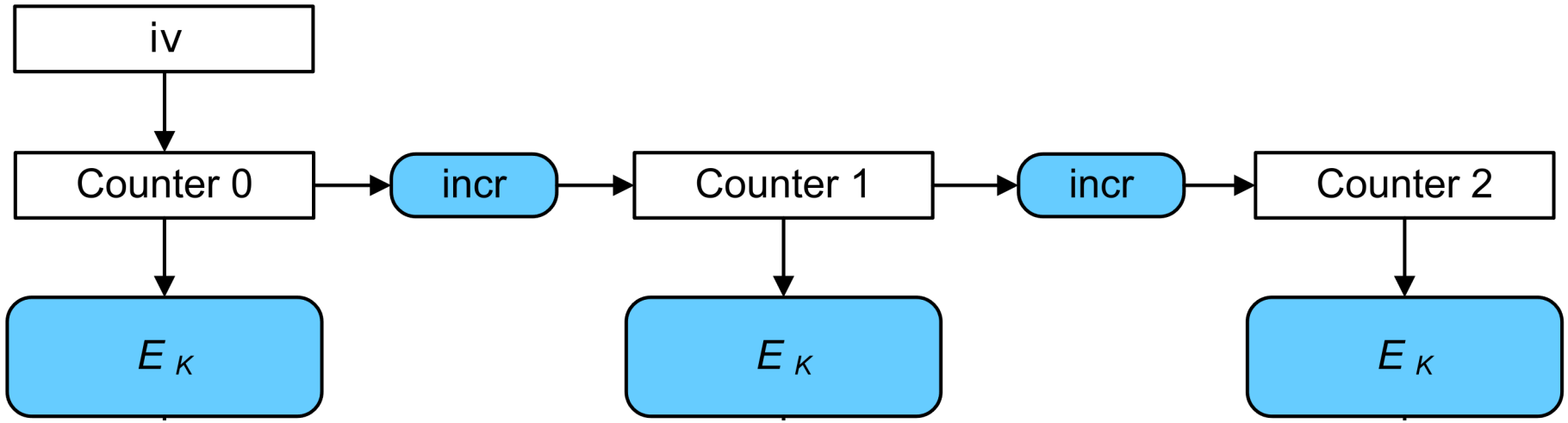
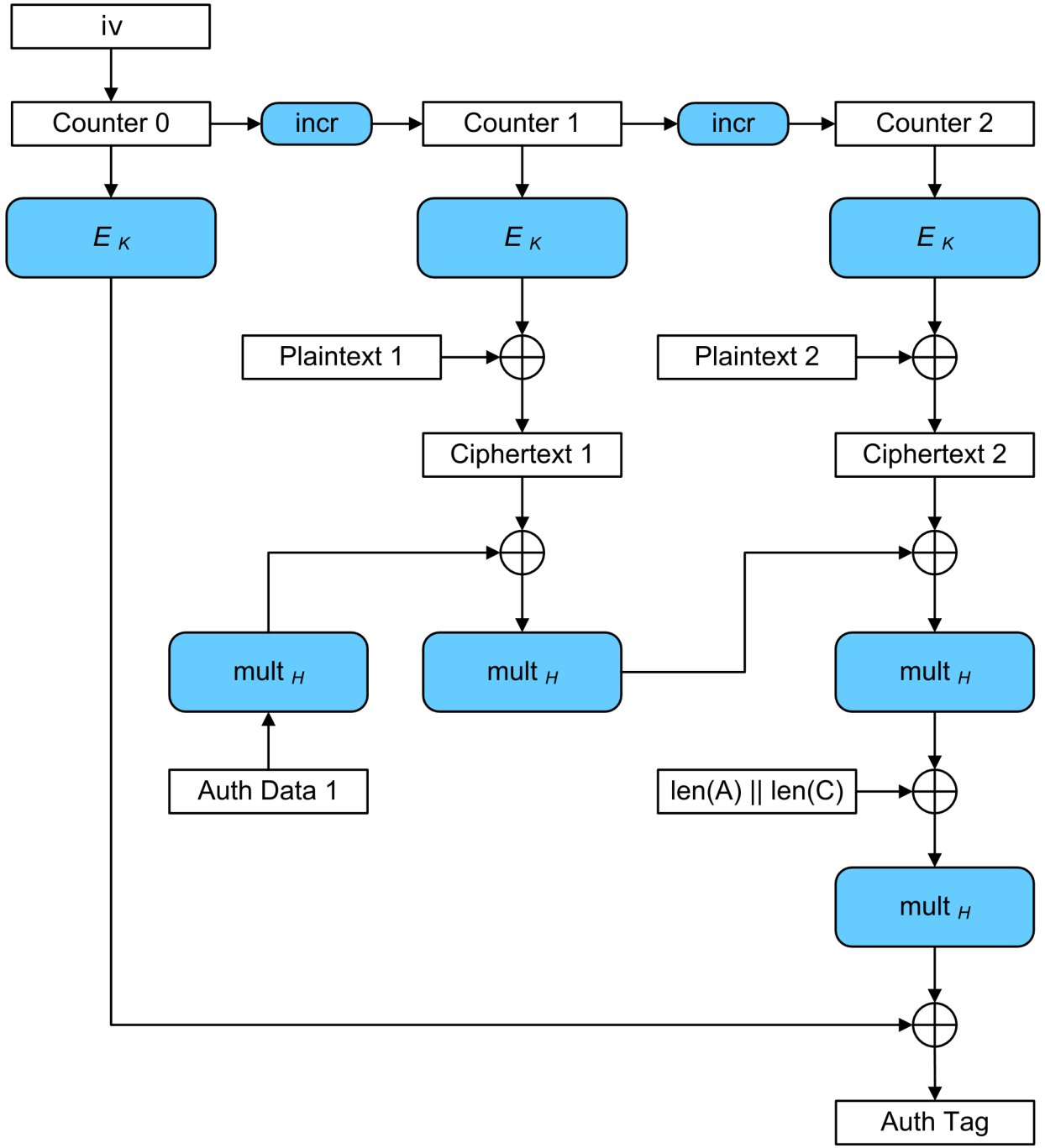


図3-4-4: GCMモード(暗号化)

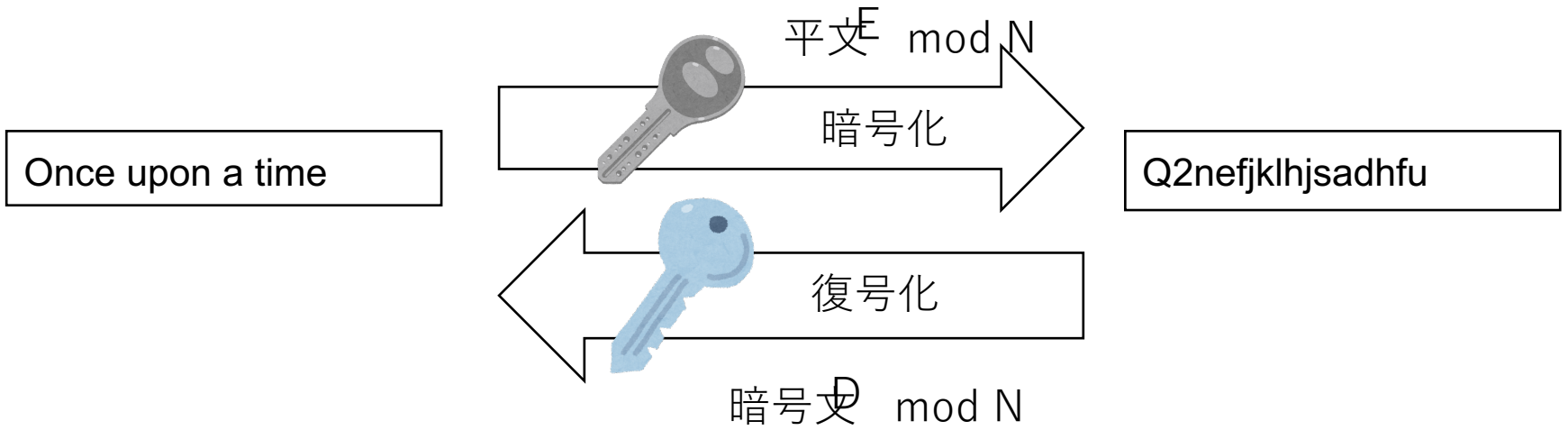
CTRモード



GCM


iv
Counter 0
incr
Counter 1
incr
Counter 2
E K
E K
E K
Plaintext 1
Plaintext 2
Ciphertext 1
Ciphertext 2
mult H
mult H
mult H
Auth Data 1
len(A) || len(C)
mult H
Auth Tag

暗号化鍵: (E, N), 復号化鍵: (D, N)

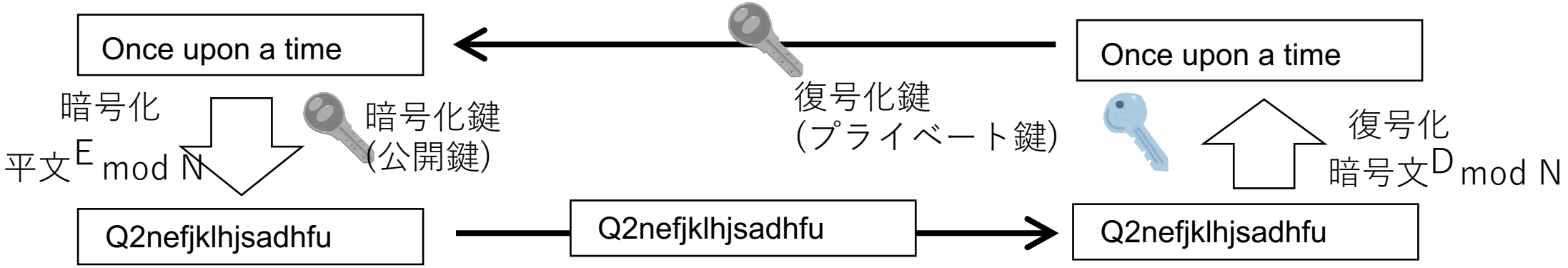


暗号化鍵: (5, 323), 復号化鍵: (29, 323)

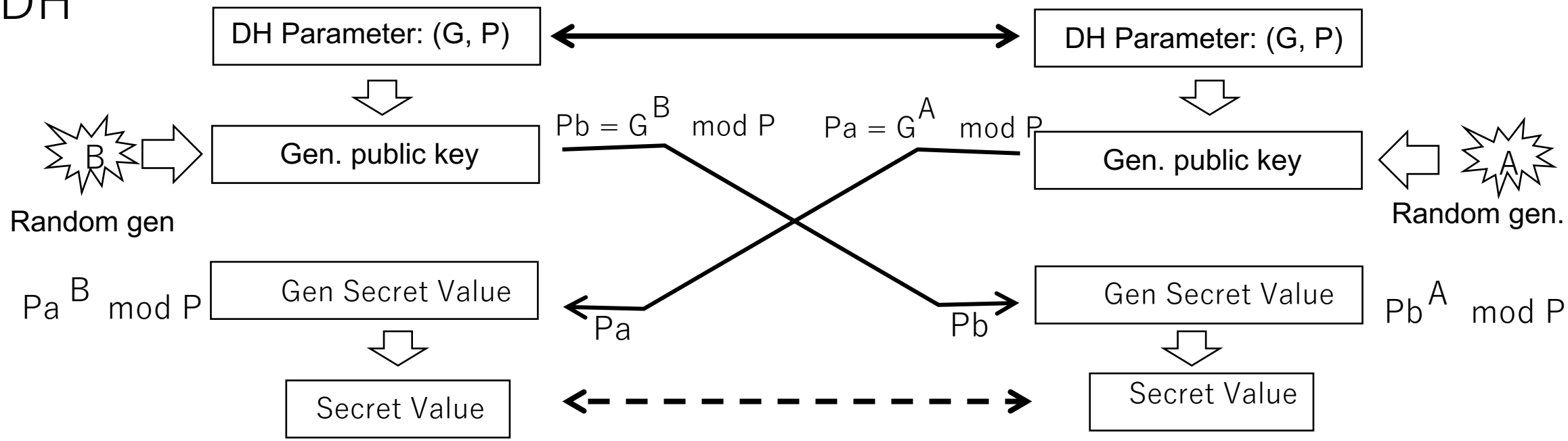
暗号化: 平文^5 mod 323

復号化: 暗号文^{29} mod 323

RSA


Once upon a time
暗号化
平文E mod N
暗号化鍵
(公開鍵)
復号化鍵
(プライベート鍵)
Once upon a time
復号化
暗号文D mod N
Q2nefjklhjsadhfu
Q2nefjklhjsadhfu
Q2nefjklhjsadhfu

DH

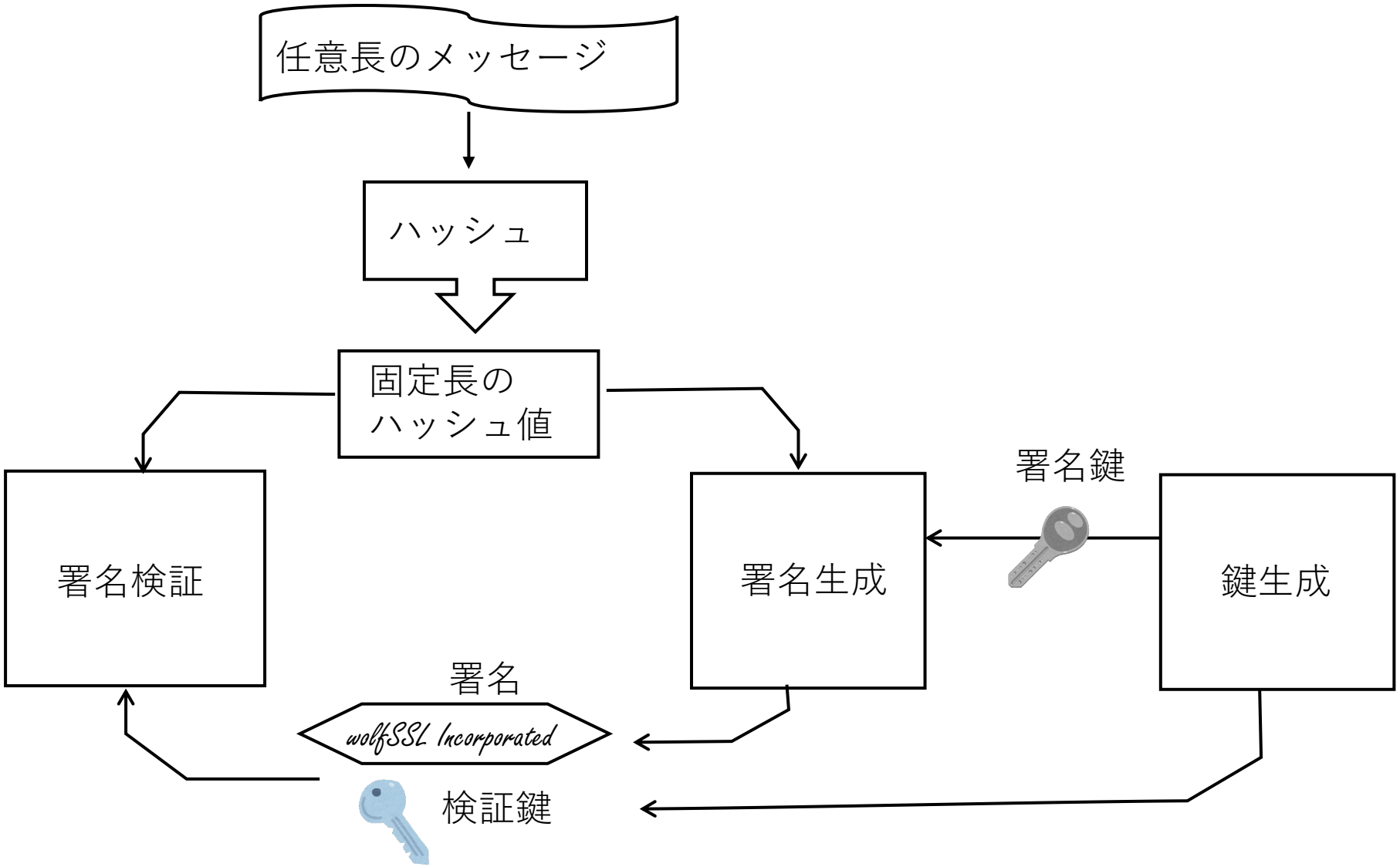

DH Parameter: (G, P)
DH Parameter: (G, P)
B
Gen. public key
Pb = G^B mod P
Pa = G^A mod P
Gen. public key
A
Random gen
Random gen.
Pa^B mod P
Gen Secret Value
Pa
Pb
Gen Secret Value
Pb^A mod P
Secret Value
Secret Value

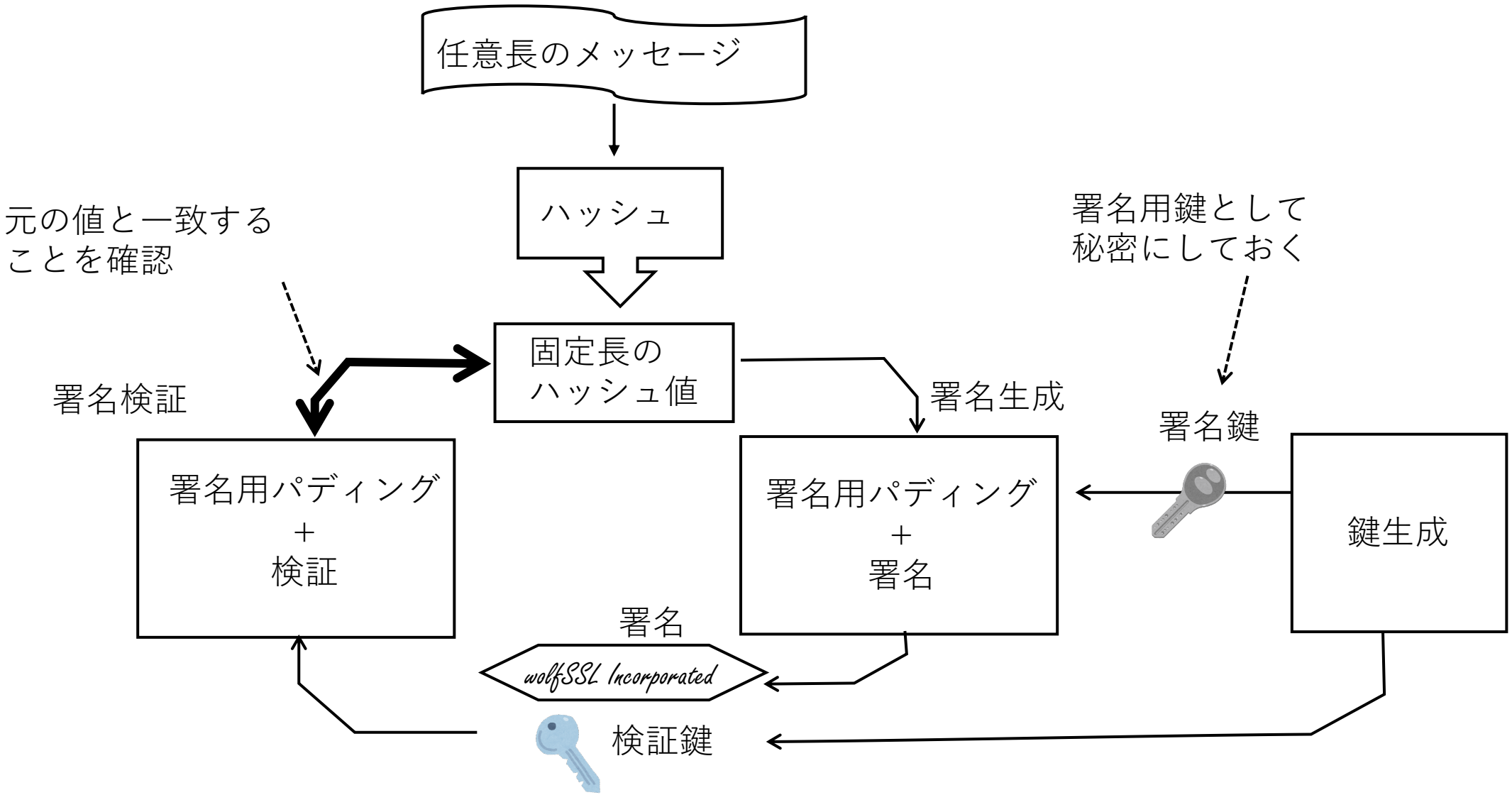
デジタル署名の目的

1) 真正性
　　メッセージと署名が正しく対応するものであることの検証
2) 否認防止
　　署名た正しい署名者によるものであることの検証

1）だけなら、共通鍵によるメッセージ認証コード(MAC)でも可能
2）は公開鍵のみで可能
- 共通鍵では、認証コード生成鍵と検証鍵が同じ
　　→　検証者が認証コードを生成できる
- 公開鍵
　　→　署名生成鍵と検証鍵が異なる。
　　秘密鍵を持ったものだけが署名可能

デジタル署名のしくみ


任意長のメッセージ
ハッシュ
固定長の
ハッシュ値
署名検証
署名生成
署名鍵
鍵生成
署名
wolfSSL Incorporated
検証鍵

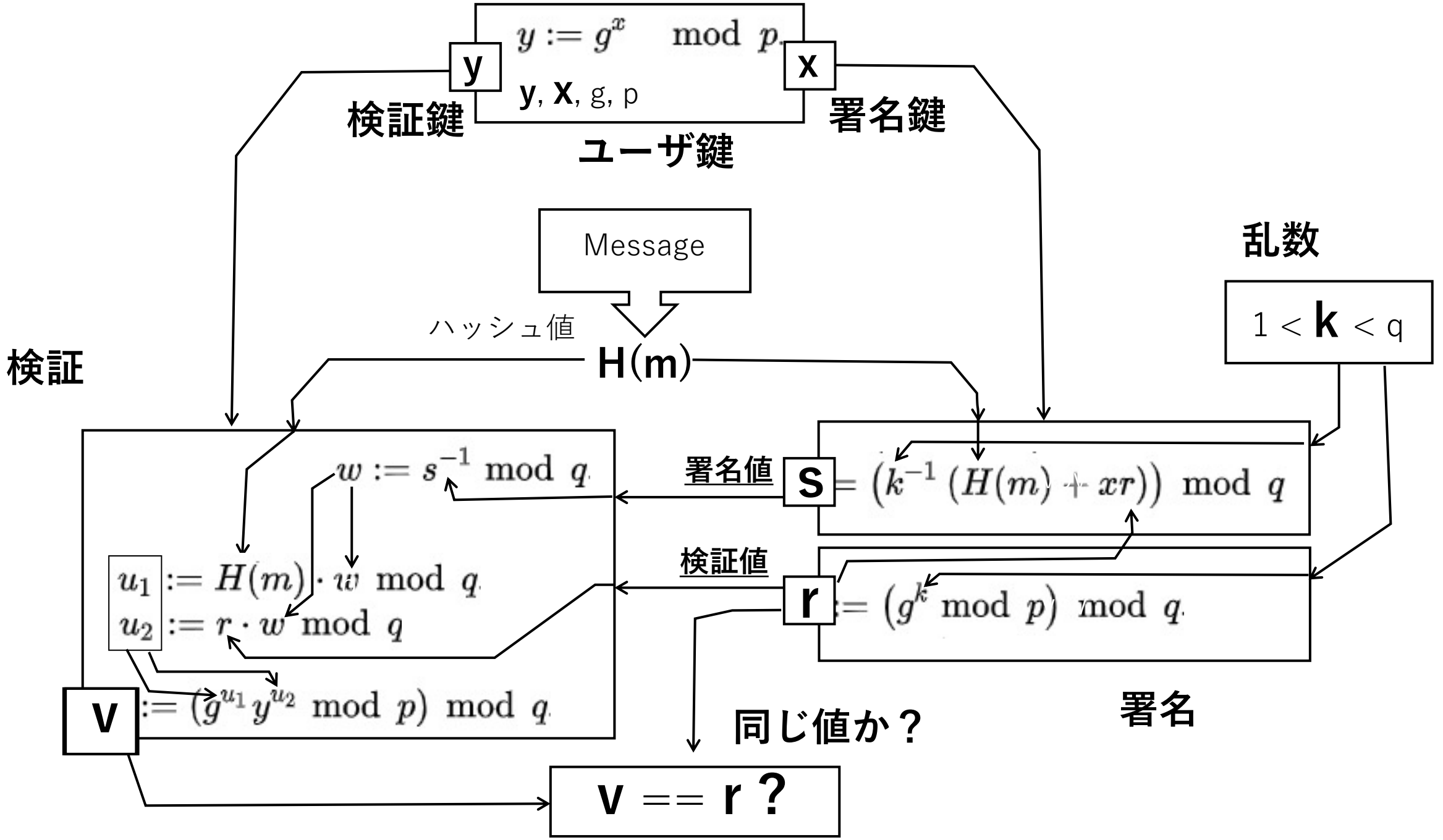

任意長のメッセージ
ハッシュ
元の値と一致する
ことを確認
署名用鍵として
秘密にしておく
固定長の
ハッシュ値
署名検証
署名生成
署名鍵
署名用パディング
＋
検証
署名用パディング
＋
署名
鍵生成
署名
wolfSSL Incorporated
検証鍵

DSA署名

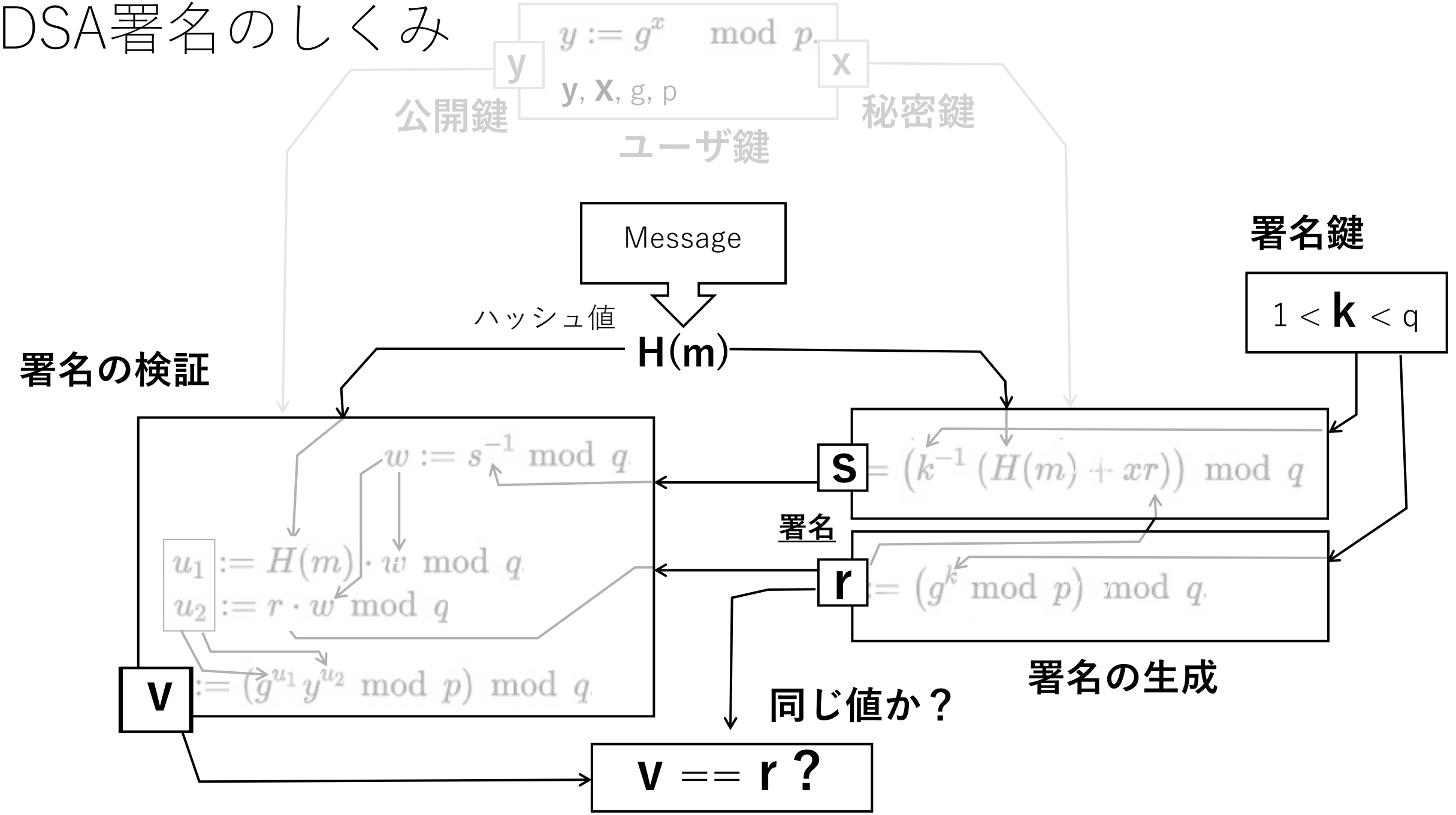
特徴：異なる二つの単方向演算の組合わせで、
同一の値を得ることができることを利用
この意味ではDHと似ている
巡回群を使っていない　　→ パディング不要

しかし、
適切な鍵生成が難しい
鍵生成を誤ると**脆弱性リスク**　→ 使用推奨されない

一方、
楕円曲線暗号では巡回群は見つかっていない
楕円曲線暗号では、上記のリスク無し
→ **ECDSA, EdDSA**として普及(ECCの章で説明)


$y := g^x \mod p$
y
x
y, x, g, p
検証鍵
署名鍵
ユーザ鍵
Message
乱数
$1 < k < q$
ハッシュ値
H(m)
検証
$w := s^{-1} \bmod q$
署名値
S
$= \left(k^{-1}\left(H(m) + xr\right)\right) \bmod q$
$u_1 := H(m) \cdot w \bmod q$
$u_2 := r \cdot w \bmod q$
検証値
r
$= \left(g^k \bmod p\right) \bmod q$
v
$:= \left(g^{u_1} y^{u_2} \bmod p\right) \bmod q$
署名
同じ値か？
v == r ?

DSA署名のしくみ


$y := g^x \mod p$
y
x
y, x, g, p
公開鍵
秘密鍵
ユーザ鍵
Message
ハッシュ値
H(m)
署名鍵
1 < k < q
署名の検証
$w := s^{-1} \mod q$
$u_1 := H(m) \cdot w \mod q$
$u_2 := r \cdot w \mod q$
$v := (g^{u_1} y^{u_2} \mod p) \mod q$
v
s
$= (k^{-1}(H(m) + xr)) \mod q$
署名
r
$= (g^k \mod p) \mod q$
署名の生成
同じ値か？
v == r ?

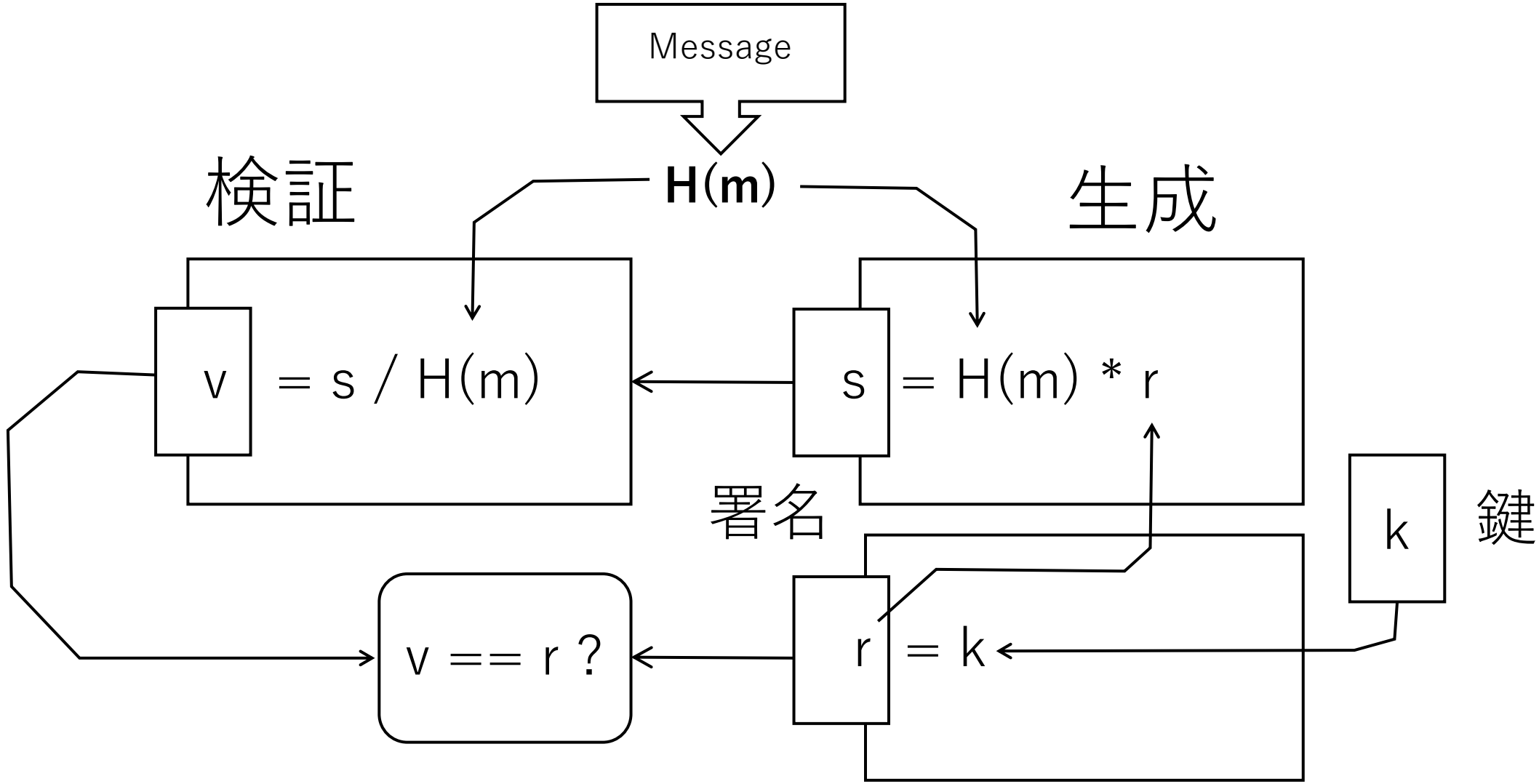
DSAの原理

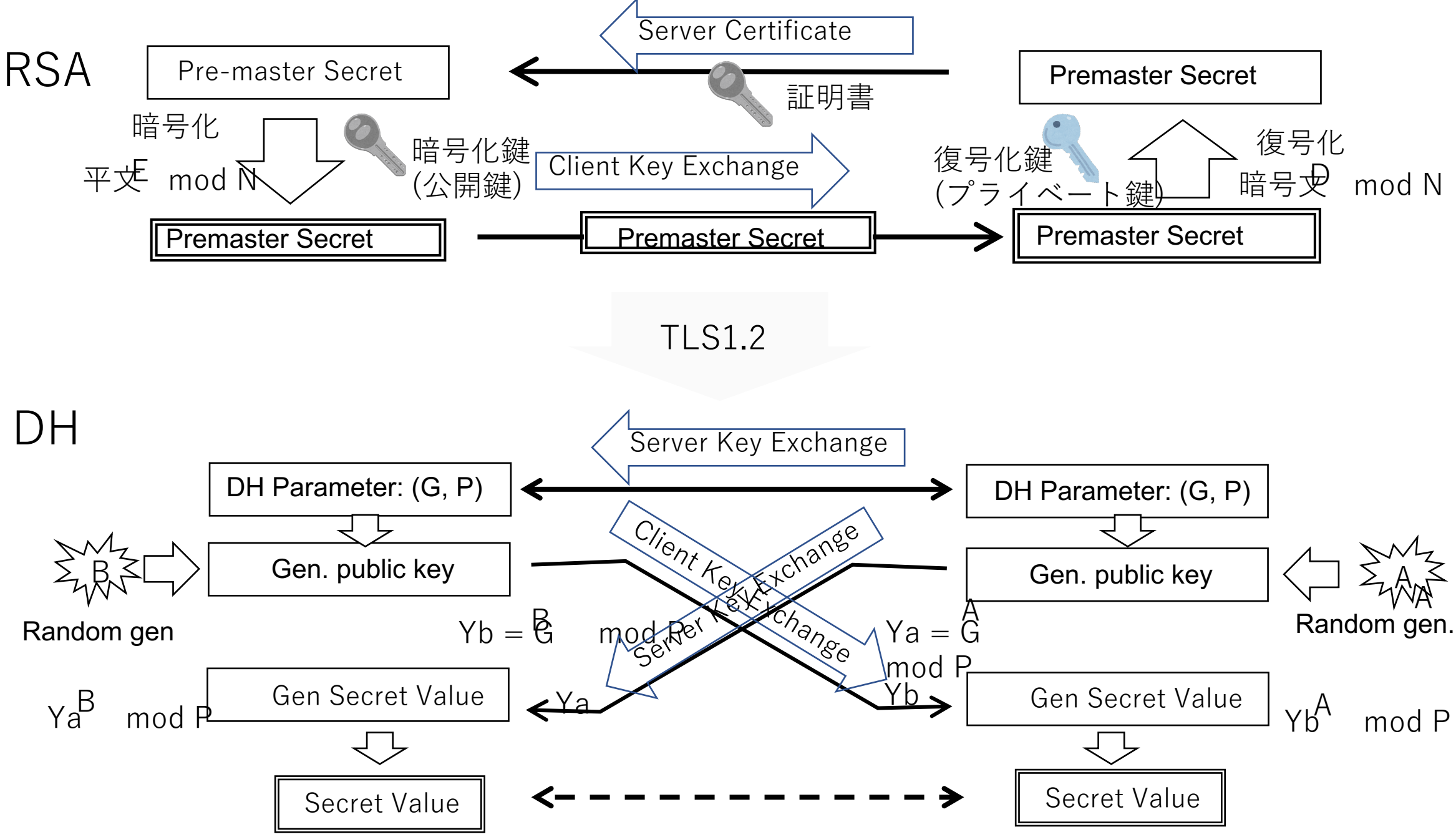
- 署名生成：H(m)と署名鍵kを入力として署名値rとsを得る。
- 値rは鍵kから求めるべき乗の剰余。公開してもkの秘匿性を保てる。
- 値sは鍵kとH(m)とから得て、検証の流れに引き継がれる。

- 検証では、値sとH(m）を入力として値vを得る。値vの計算ではH(m)が消し込まれ、結果は署名生成側の値rと同じ値になる。

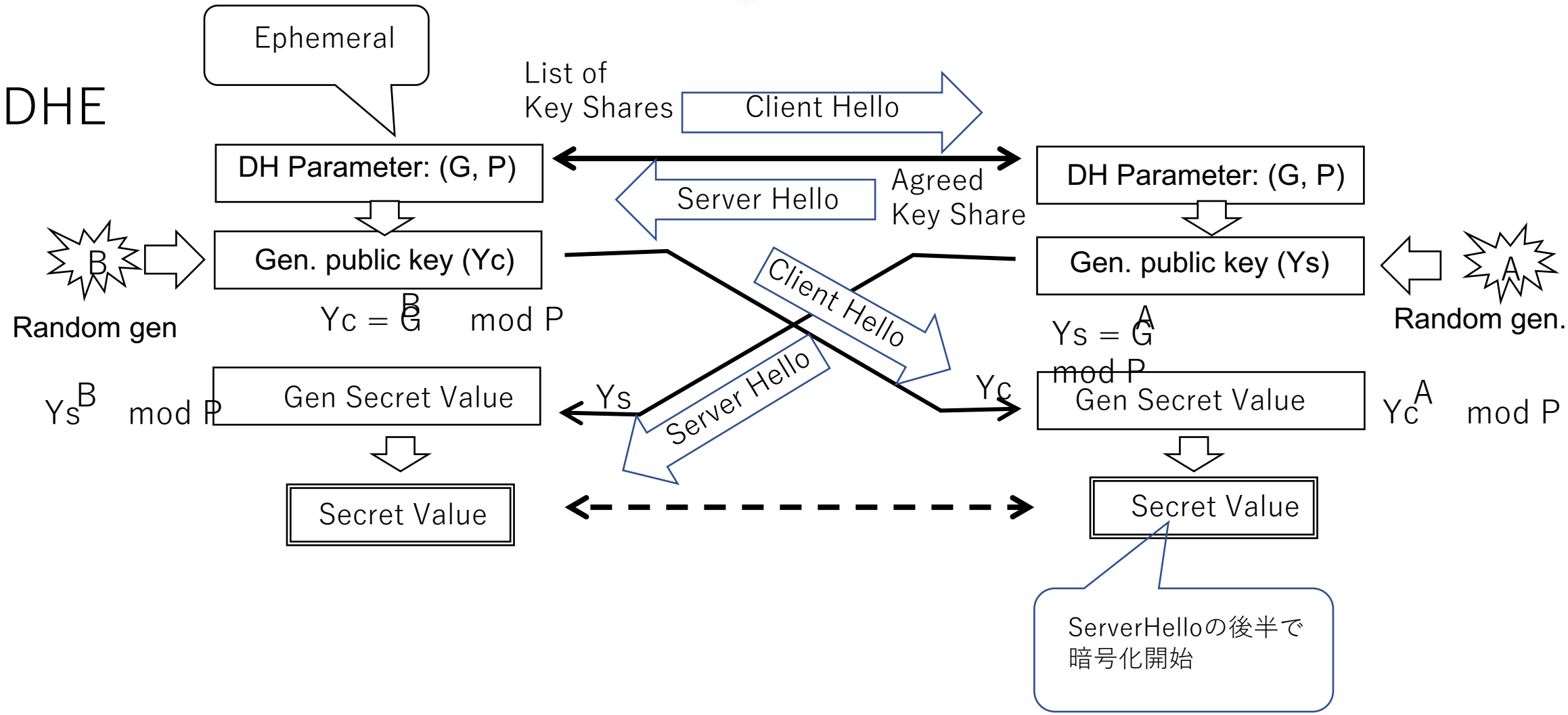
改ざんの検出：署名の生成側、検証側の双方の演算途中でH(m)が使われているため、署名の生成、検証の双方で一環したH(m)が与えられないと正しい値vが得られない。

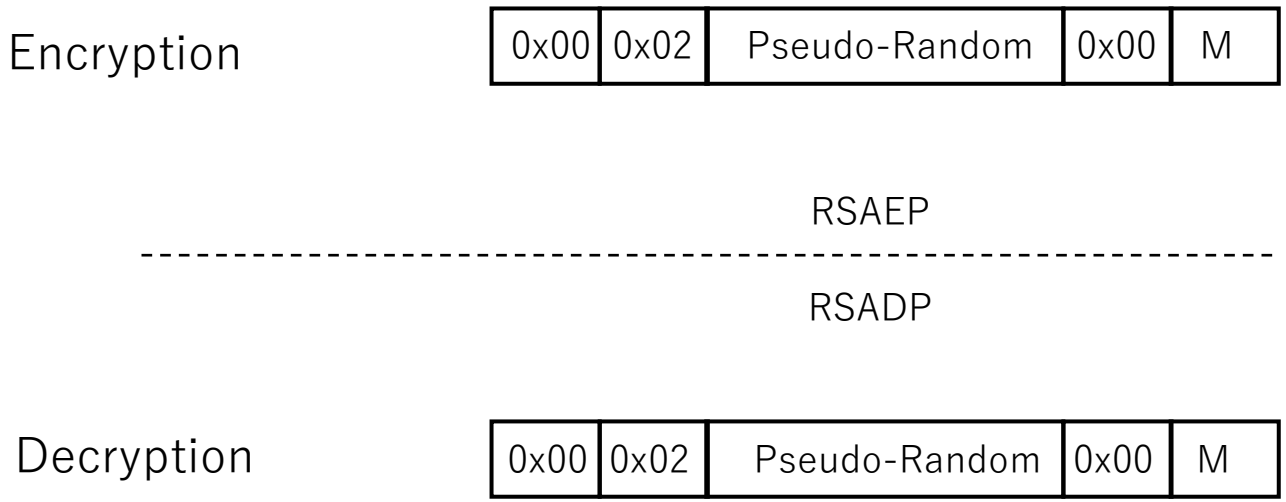
超絶簡単版DSAもどき



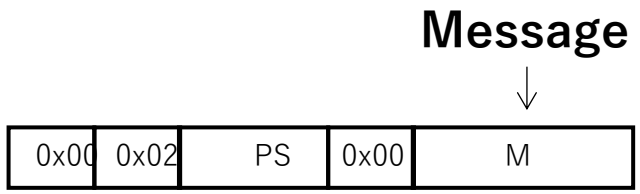

RSA
Pre-master Secret
Server Certificate
証明書
Premaster Secret
暗号化
平文 E mod N
暗号化鍵
(公開鍵)
Client Key Exchange
復号化鍵
(プライベート鍵)
復号化
暗号文 D mod N
Premaster Secret
Premaster Secret
Premaster Secret
TLS1.2
DH
Server Key Exchange
DH Parameter: (G, P)
DH Parameter: (G, P)
Client Key Exchange
Server Key Exchange
B
Gen. public key
Gen. public key
A
Random gen
Yb = G B mod P
Ya = G A mod P
Random gen.
Gen Secret Value
Gen Secret Value
Ya
Yb
Ya B mod P
Yb A mod P
Secret Value
Secret Value

TLS1.3


DHE
Ephemeral
List of
Key Shares
Client Hello
DH Parameter: (G, P)
Server Hello
Agreed
Key Share
DH Parameter: (G, P)
B
Gen. public key (Yc)
Gen. public key (Ys)
A
Random gen
Yc = G^B mod P
Ys = G^A mod P
Random gen.
Client Hello
Server Hello
Ys^B mod P
Gen Secret Value
Ys
Yc
Gen Secret Value
Yc^A mod P
Secret Value
Secret Value
ServerHelloの後半で
暗号化開始

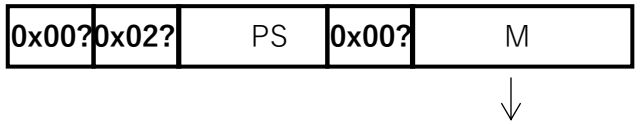

Encryption
0x00
0x02
Pseudo-Random
0x00
M
RSAEP
RSADP
Decryption
0x00
0x02
Pseudo-Random
0x00
M

Encryption



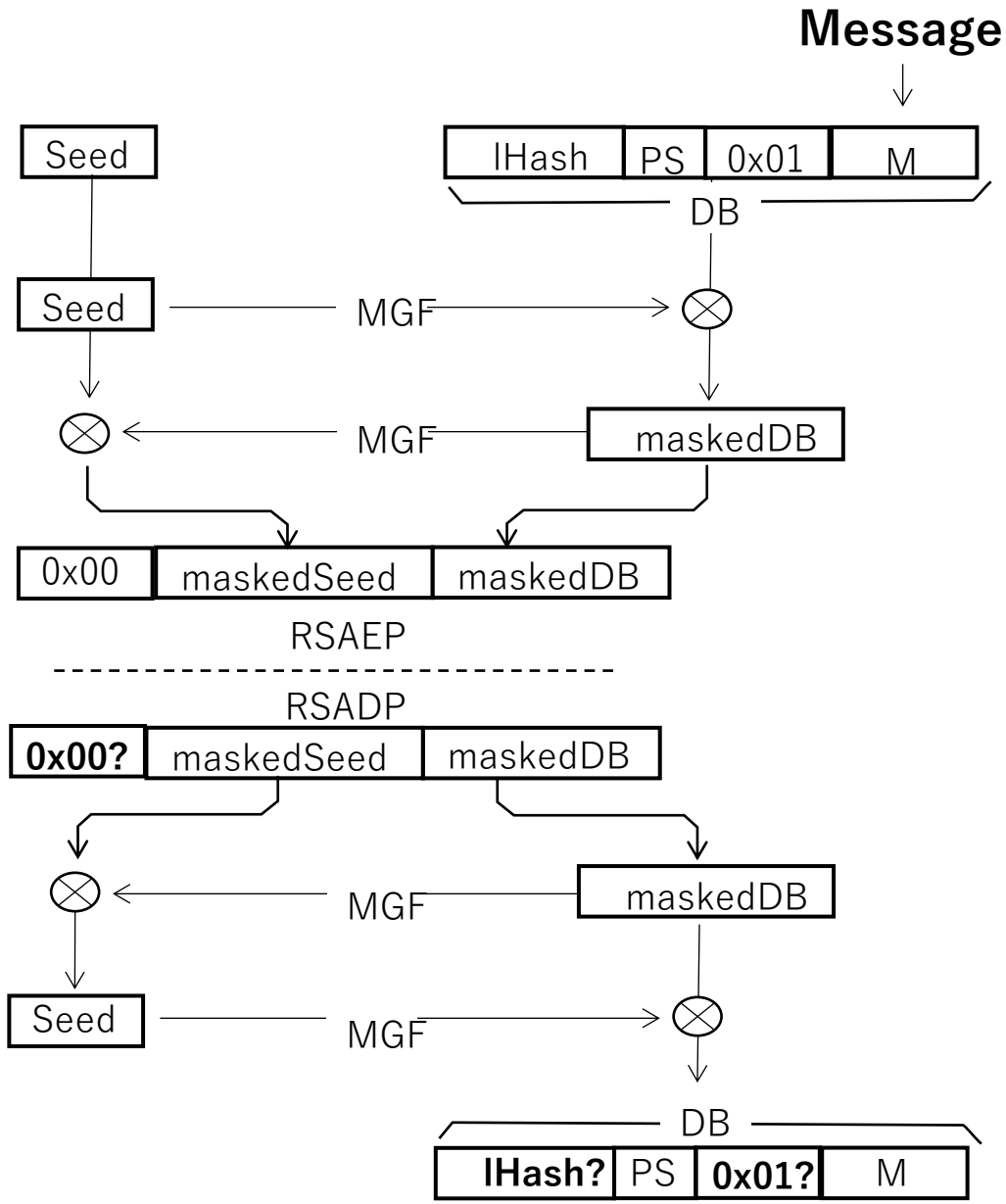
RSAEP

RSADP



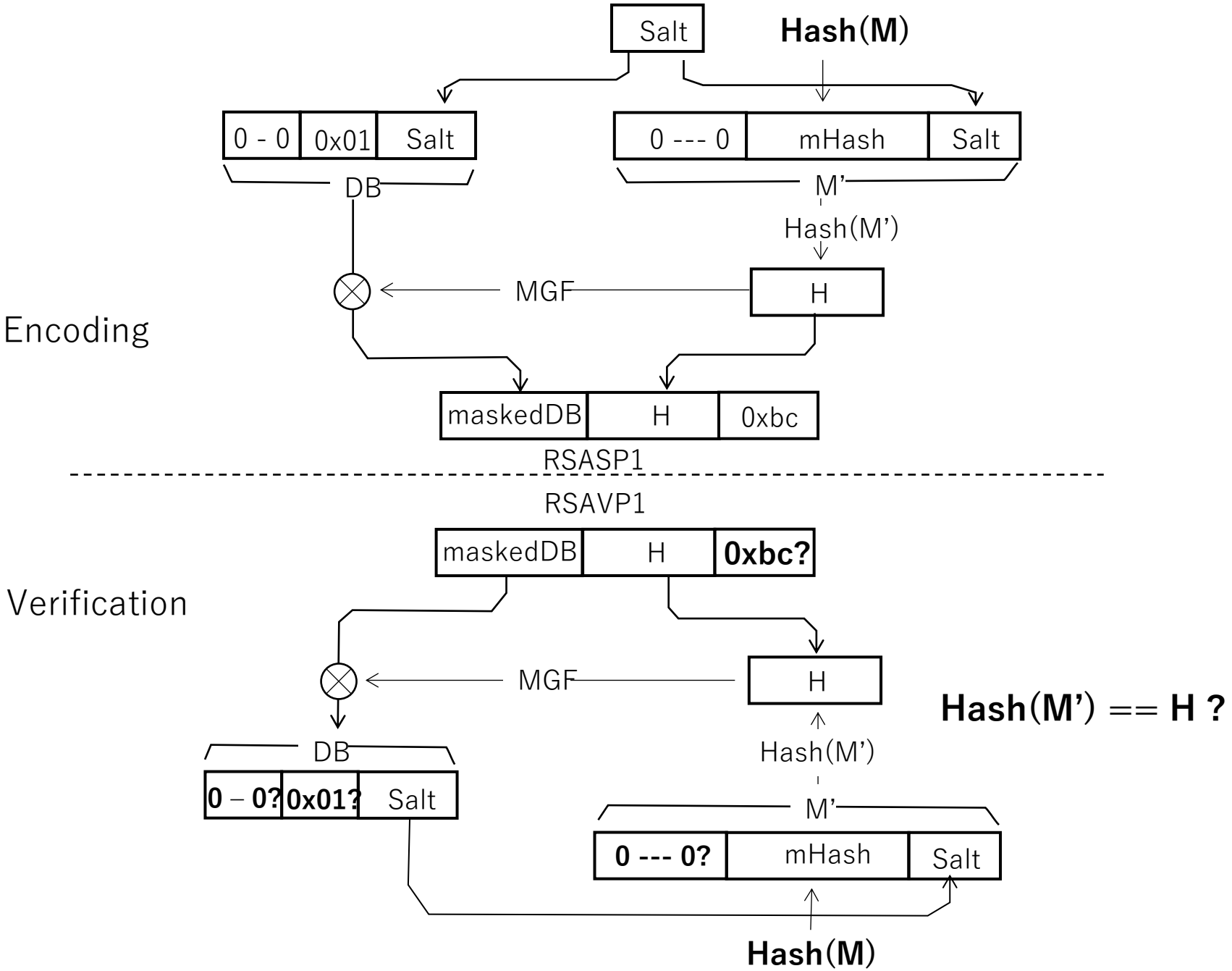
Decryption

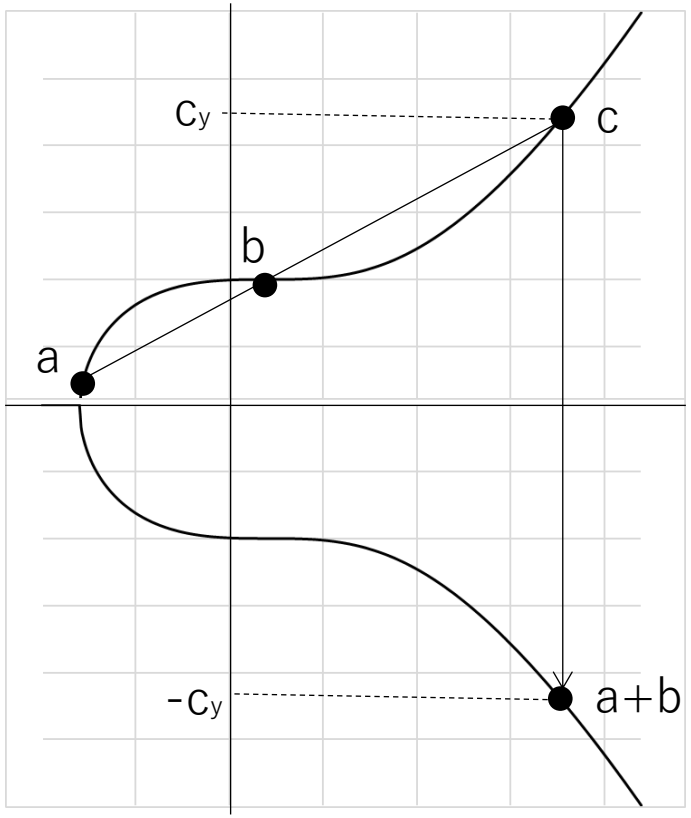
PKCS#1 V1.5



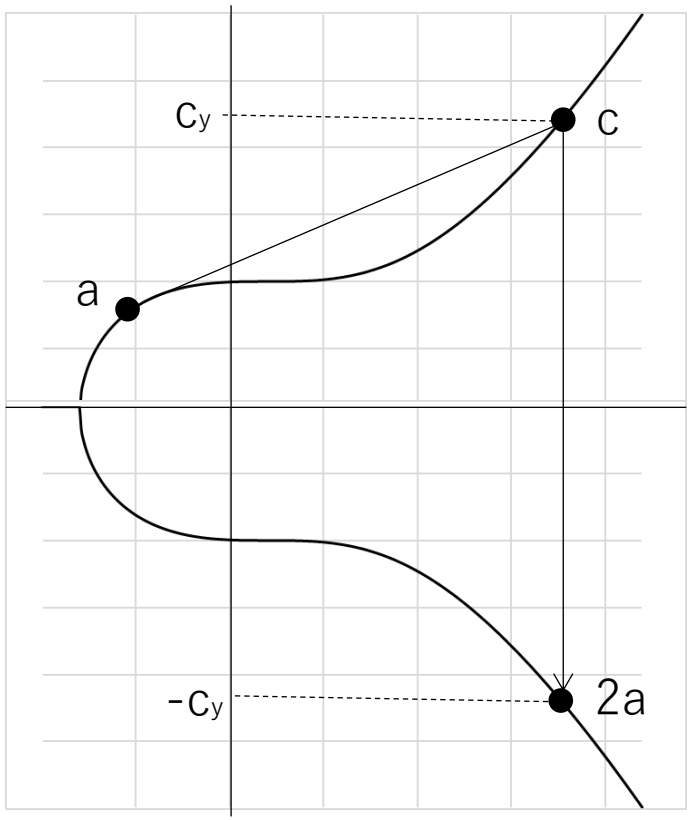
OAEP

PS: Pseudo-Random

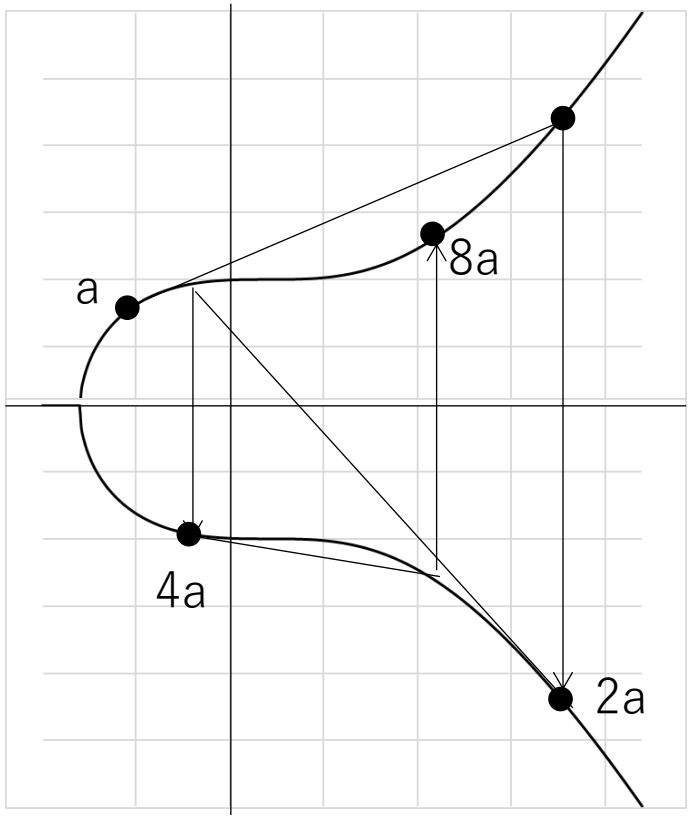

Salt
Hash(M)
0 - 0
0x01
Salt
DB
0 --- 0
mHash
Salt
M'
Hash(M')
MGF
H
Encoding
maskedDB
H
0xbc
RSASP1
RSAVP1
maskedDB
H
0xbc?
Verification
MGF
H
Hash(M') == H ?
DB
0 – 0?
0x01?
Salt
Hash(M')
M'
0 --- 0?
mHash
Salt
Hash(M)



点aと点bの加算



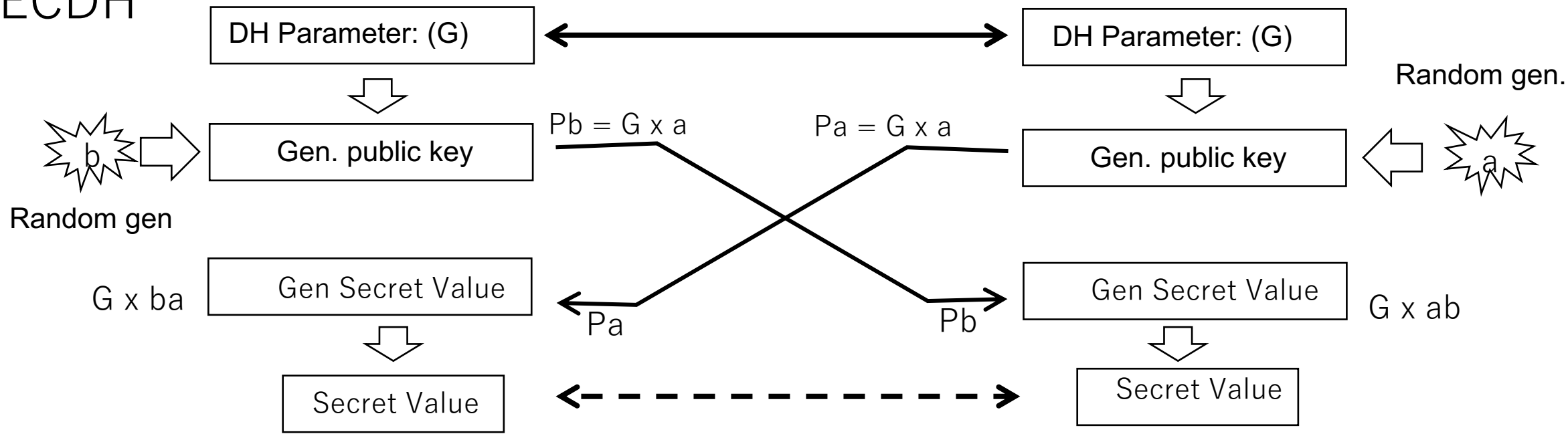
点a×2

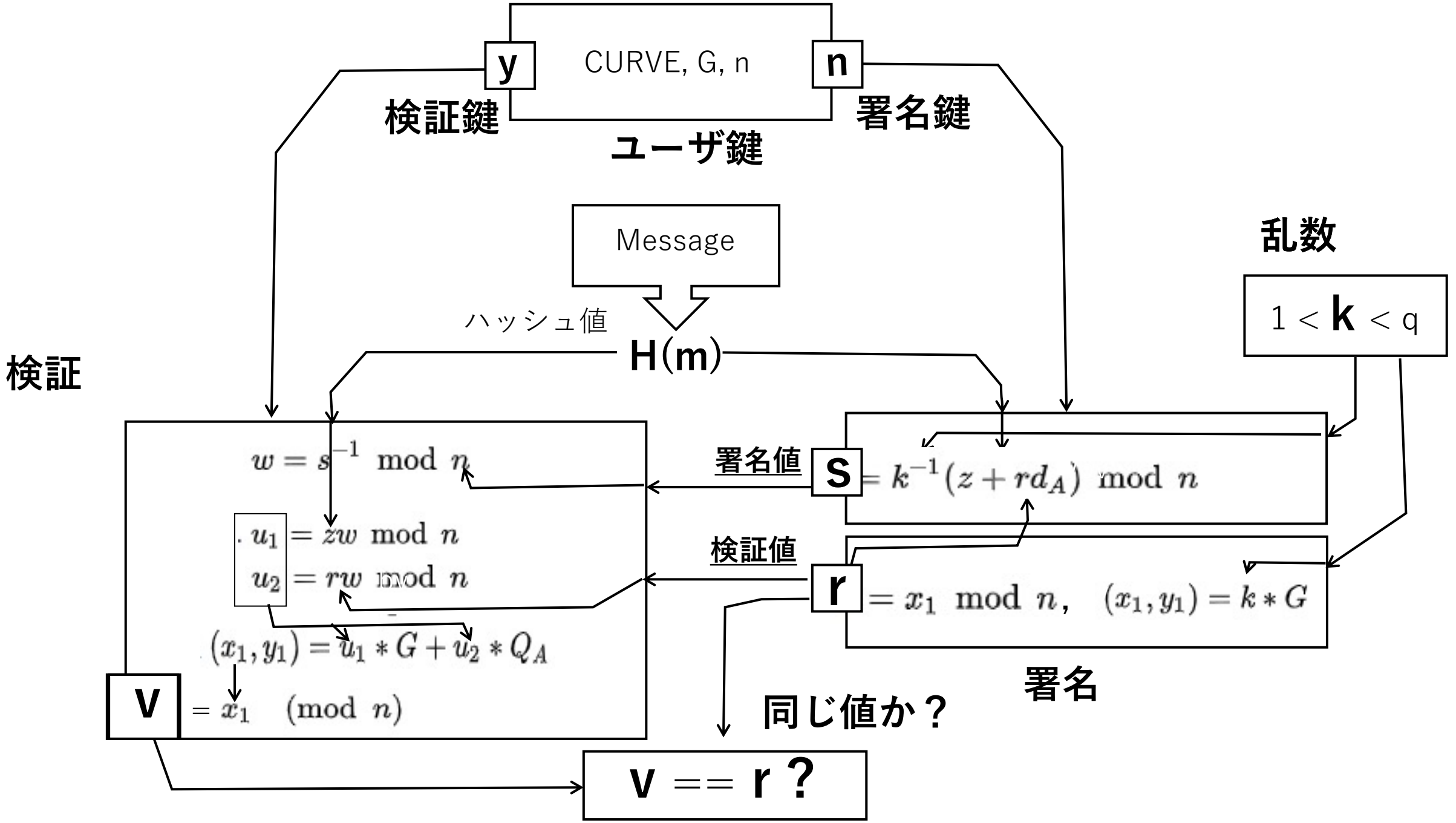


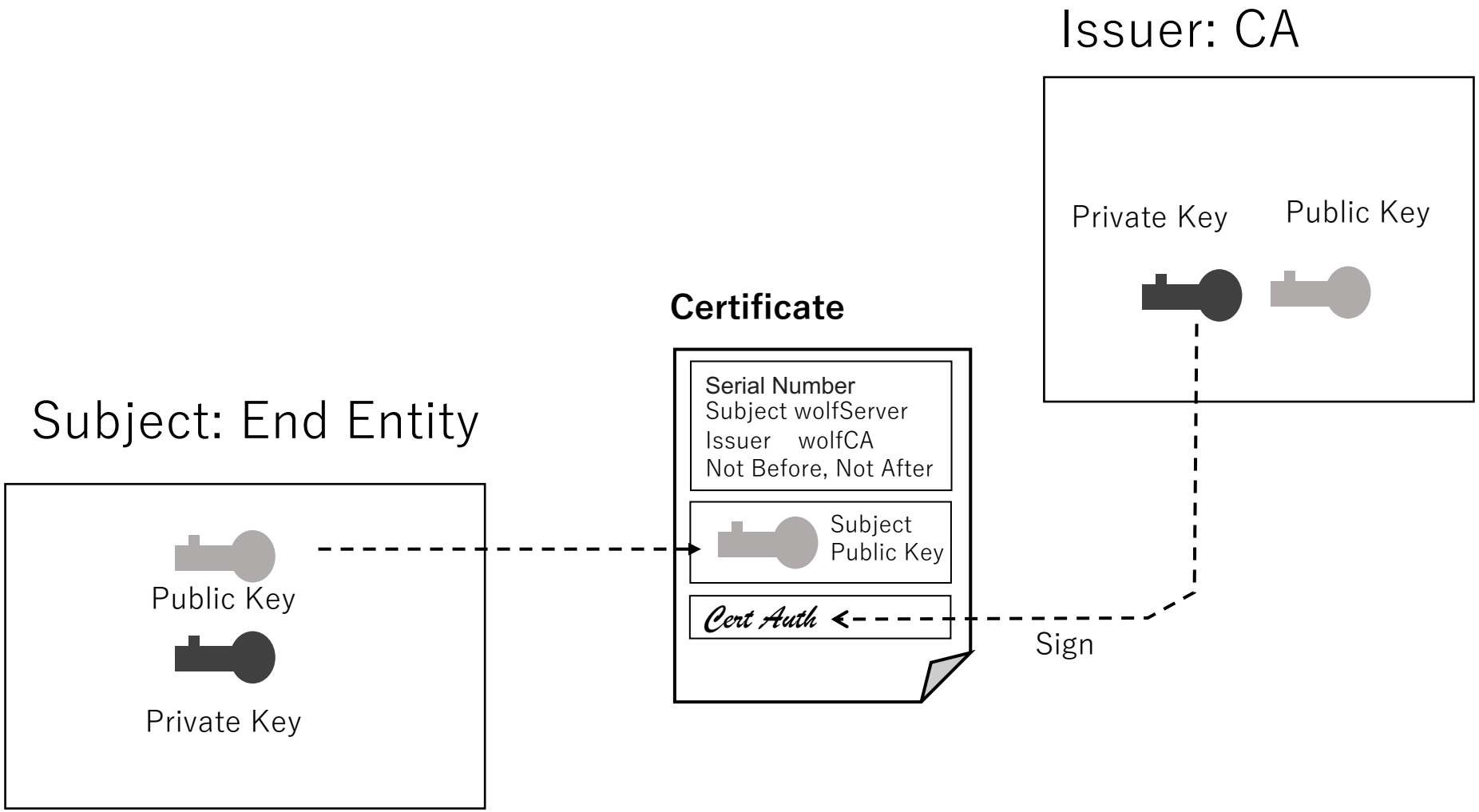
点a×2^n

楕円曲線上の演算を定義する

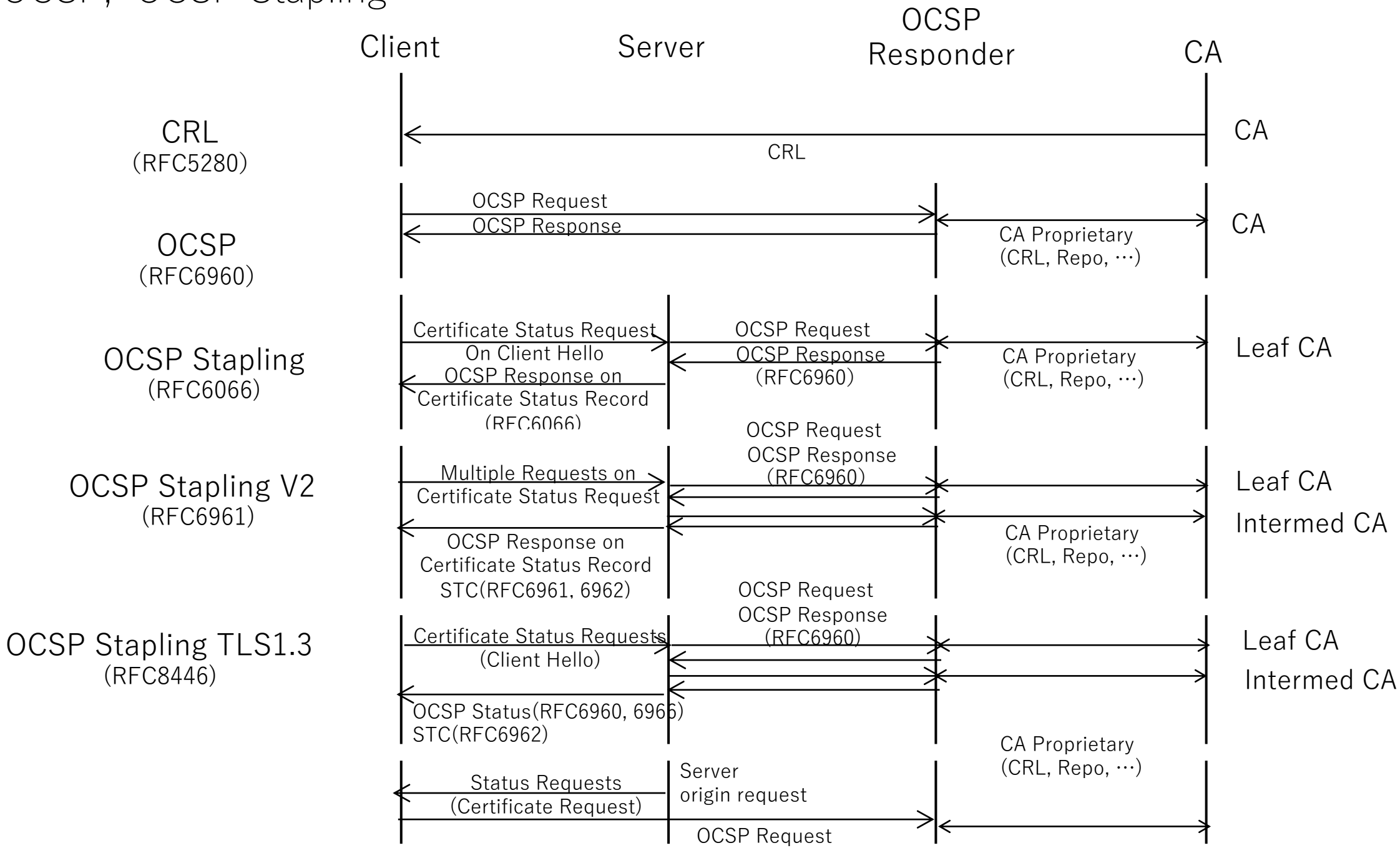
ECDH

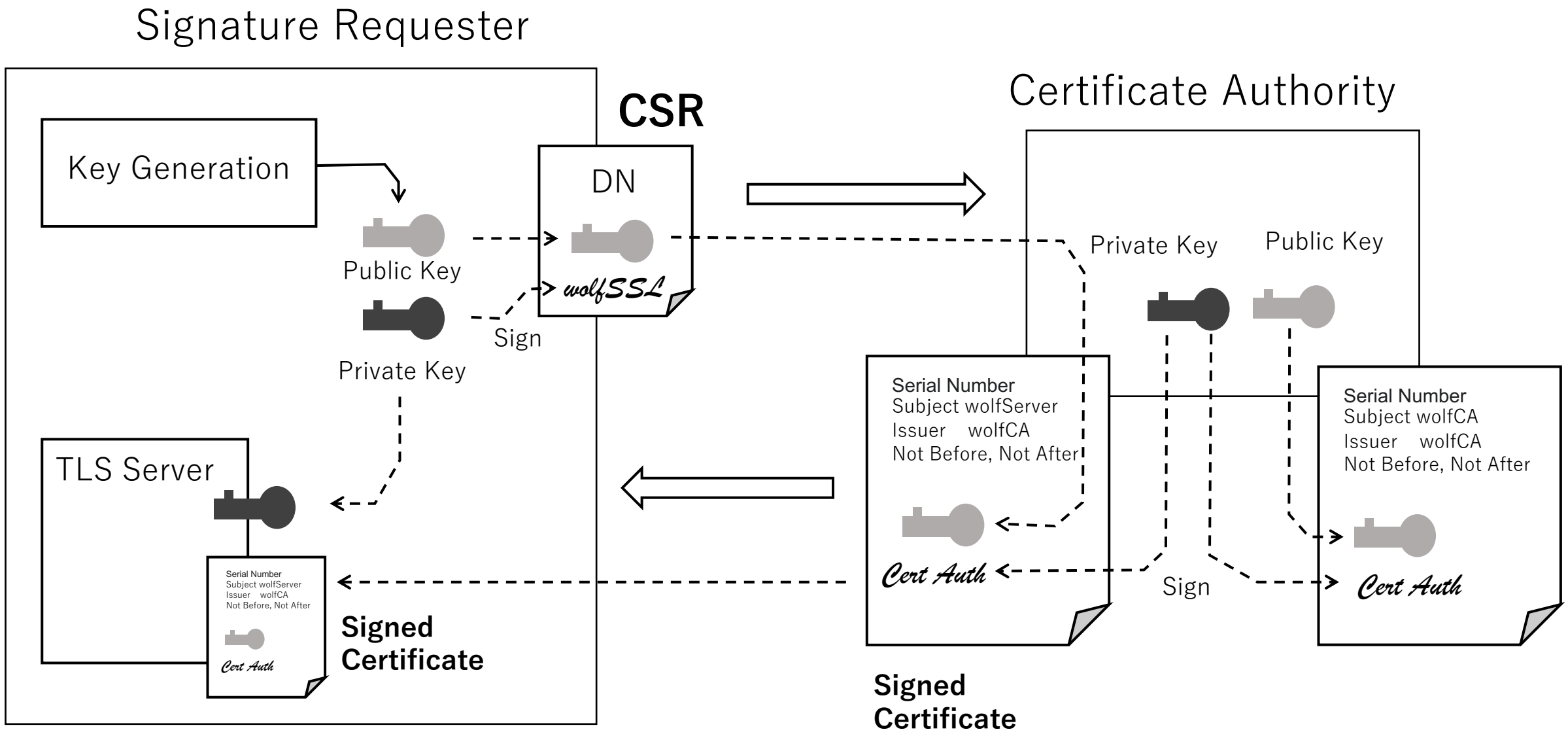

DH Parameter: (G)
DH Parameter: (G)
Random gen.
b
Gen. public key
Pb = G x a
Pa = G x a
Gen. public key
a
Random gen
G x ba
Gen Secret Value
Pa
Pb
Gen Secret Value
G x ab
Secret Value
Secret Value


y
CURVE, G, n
n
検証鍵
署名鍵
ユーザ鍵
Message
ハッシュ値
H(m)
乱数
1 < k < q
検証
$w = s^{-1} \mod n$
署名値
S
$= k^{-1}(z + rd_A) \mod n$
$u_1 = zw \mod n$
$u_2 = rw \mod n$
検証値
r
$= x_1 \mod n, \quad (x_1, y_1) = k * G$
$(x_1, y_1) = u_1 * G + u_2 * Q_A$
V
$= x_1 \pmod n$
署名
同じ値か？
V == r ?


Issuer: CA
Private Key
Public Key
Certificate
Serial Number
Subject wolfServer
Issuer wolfCA
Not Before, Not After
Subject
Public Key
Cert Auth
Sign
Subject: End Entity
Public Key
Private Key

CRT, OCSP, OCSP Stapling




Signature Requester
Key Generation
Public Key
Private Key
CSR
DN
wolfSSL
Sign
Certificate Authority
Private Key
Public Key
Serial Number
Subject wolfServer
Issuer wolfCA
Not Before, Not After
Cert Auth
Serial Number
Subject wolfCA
Issuer wolfCA
Not Before, Not After
Cert Auth
Sign
Signed Certificate
TLS Server
Serial Number
Subject wolfServer
Issuer wolfCA
Not Before, Not After
Cert Auth
Signed Certificate