平成 13 年度

学士学位論文

SAS プロトコルにおける同期問題

A synchronous problem in SAS protocol

1020277　　上岡 隆

指導教員　　清水 明宏

2002 年 2 月 8 日

高知工科大学 情報システム工学科

要 旨

SASプロトコルにおける同期問題

上岡 隆

情報通信分野の発展に伴い，インタネットやモバイルコミュニケーション環境において，通信相手の資格認証が必要不可欠なものとなっている．本研究室では，セキュリティが十分でないネットワーク上でユーザ等の資格認証を行う際，被認証者側および認証者側共に，実行する計算量が極めて少なく，被認証者側にも認証者側にも簡易で小さいプログラムサイズで実現可能，さらに，通信路上での盗聴に強い安全な認証方式，SASパスワード認証方法を提案している．本方式は現在，携帯電話における個人認証用に実装されているが，この実用システムにおいて用いられている同期管理方式には，第三者の改ざんにより認証が不能になる問題がある．

本論文では，SASプロトコルにおける既存の同期管理方式における問題点を指摘し，ほとんど追加コストを必要とせずにこの問題を解決する方法を提案・評価する．

キーワード　SAS認証方式，ワンタイムパスワード，同期管理，盗聴，サービス不能攻撃

Abstract

A synchronous problem in SAS protocol

Takashi KAMIOKA

As information communication technology advances, it has become of extreme importance to authenticate the user and the other party communicated with, in Internet and mobile communication. Our laboratory have proposed the SAS (Simple And Secure) password authentication method for networks that are not sufficiently secure. This features a low processing load on both the authenticated and authenticating sides, with only a small, simple program required at both sides. In addition, it provides strong protection against tapping on the communication path. There is a problem to which authentication becomes impossible by a third person's alteration in the synchronous management system used in this practical use system.

In this paper, I will shown the problem in the existing synchronous management system in an SAS protocol, and propose and evaluate the method of solving this problem, without hardly needing additional cost.

key words SAS Authenticaton protocol, one-time passward, Synchronous management, tapping, Denial of Service Attack

目次

図目次

第 1 章

はじめに

インタネットおよびモバイルコミュニケーション環境において，ユーザや通信相手の資格認証が必要不可欠なものとなっている．資格認証方法として，(1) 指紋や顔，声等の身体情報を利用したもの，(2) IC カードのような本人の所有物を利用したもの，(3) パスワードや暗証番号等の本人の知識によるものが考えられる．しかし，(1)，(2) は指紋や顔，声，IC カード等を読み込む装置等が必要であり，インタネットやモバイルコミュニケーション環境で利用するにはコストが大きく，現実的でない．

そこでこれらの環境でよく用いられるのが (3) のパスワードを用いた資格認証方法であり，特に，毎回送信される認証データが異なるために通信の盗聴に強いワンタイムパスワード認証方式が広く用いられている [1][2][3]．しかし，Lamport の方法には，被認証者がパスワードを定期的に再設定する必要性や，認証者側に比較して処理能力の小さい被認証者側での処理負担が大きいという問題がある．また，CINON 法では，Lamport の方法の欠点であるパスワードの更新の必要性をなくす事ができたが，被認証者および認証者における計算量が大きいという問題は依然残った．さらに，通信途中の認証情報を第三者により二回続けて盗聴された場合,「なりすまし」の危険性もあった．

そこで我々はセキュリティが十分でないネットワーク上でユーザ等の資格認証を行う際，被認証者側および認証者側共に，実行する計算量が極めて少なく，被認証者側にも認証者側にも簡易で小さいプログラムサイズで実現可能，さらに，通信路上での盗聴に強い安全な認証方式，SAS パスワード認証方法を提案した [4]．本方式は現在，携帯電話での個人認証用に実装されているが，実装されているカウンタ同期方式には通信途中で第三者に認証情報をすり替えられる事によって正規ユーザが以降の認証を行えなくなるという問題がある事を発

見した．

本論文では，SAS におけるカウンタ同期方式の問題点及び攻撃法を指摘し，ほとんど追加コストを必要とする事なく，それらの問題を克服できる案を提案する．

本論文の以下の部分では，まずワンタイム認証方式おける SAS の優位性と同期問題，既存同期管理方式等の研究背景について **2** で述べ，カウンタ同期方式に対する脅威について **3** で，その問題の解決法と評価を **4** で述べる．最後に **5** で本論文のまとめとする．

第 2 章

研究背景

まず本章では，本研究で対象としている SAS 認証方式がワンタイムパスワードの中で優位な方式である事を示し，その SAS における同期問題と提案されている同期管理方式についてを述べる．

2.1 ワンタイムパスワード

ネットワークを経由したサーバとクライアント間の操作で，もっとも盗聴されると危険な情報はログインの情報である．そのため，パスワードが盗めないようにするか，もしくは盗まれても使えないようにする必要がある．そのための手法として，ワンタイムパスワードという手法が提案されている．ワンタイムパスワードとは，認証の際にパスワードを毎回変えるシステムである．同じパスワードを二度と使わないので，盗聴などでパスワードを知られてしまっても問題がない．以下にワンタイムパスワードの代表的なものである Lamport の方法，CINON 法，SAS について説明する．

2.1.1 定義と記法

本論文で用いる定義と記法は以下の通りである．

1. User を，認証用プロトコルを用いるコンピュータユーザとする．
2. Host を，ユーザを認証するサーバとする．
3. A を，ユーザの ID とする．
4. S を，ユーザのパスワードとする．

5. n を，認証セッション回数を表す 0 以上の整数とする．
6. N_n を，n 回目の認証に対応する乱数とする．
7. P を，サーバの識別のための公開情報とする．
8. M_n は認証子とする．
9. E を，暗号用のハッシュ関数とする．また E_n^m とは，N_n を使い，$(S \oplus N_n)$ が m 回ハッシュされた事を示す．
10. $\oplus$ は，ビットごとの排他的論理和を表す．

2.1.2 Lamport の方法

Lamport の方法の認証メカニズムは，一方向性関数を複数回適用したデータを事前にサーバ側に登録しておき，認証の際には，登録しているデータの適用一回前のデータを示す事によって認証を行い，次回用の鍵はユーザが示した一つ前のデータに置き換えるという方式である．概要は以下に示す (図 2.1 参照)．

(1) 登録フェーズ ($n = 0$)

User: P，m を任意に設定，以下のデータを計算する．ただし，$E^0 = E(P, S)$ である．

$E^m = E(P, E^{m-1})$ $(m = 1, \cdots)$

そして，m を保存しておく．

User: E^m をサーバに送信する．

Host: E^m を保存する．

(2) 認証フェーズ ($n = k$)

User: 記録してある m および k を読み出し，E^{m-k} を計算する．

User: E^{m-k} をサーバに送信する．

Host: 受信した E^{m-k} から E^{m-k+1} を計算する．

Host: 生成した E^{m-k+1} と前回登録されている E^{m-k+1} を比較，マッチすれば認証成功として E^{m-k} を新たに鍵として記録する．アンマッチなら認証は拒否される．

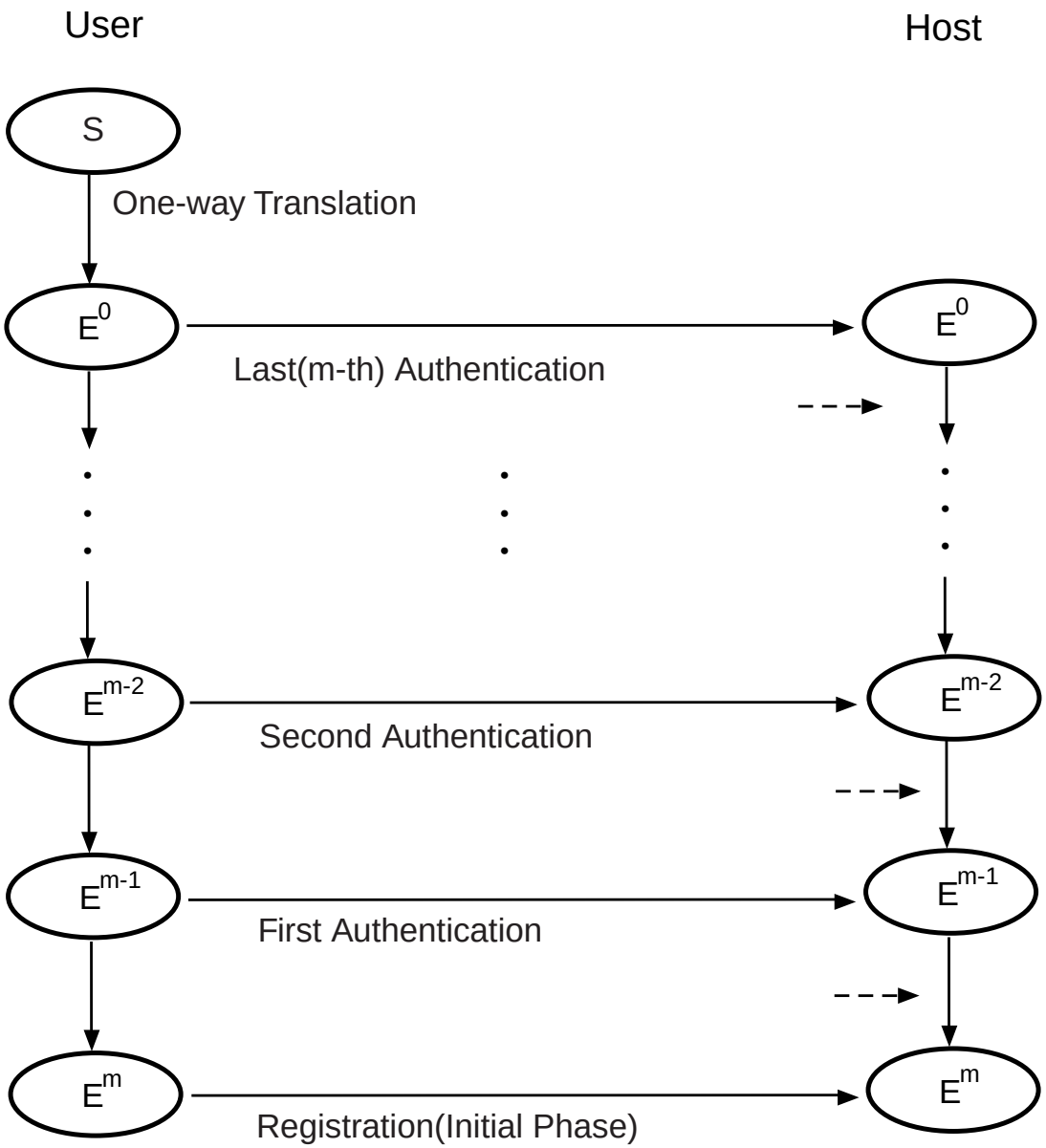

図 2.1 Lamport の方法の登録，および認証フェーズ

この方法では元のデータを公開し尽くした時点で，パスワードの再設定が必要となってしまうという問題がある．また連続認証回数を増やすために一方向性関数の適用回数を増加させると処理量が増大してしまい，サーバに比べて処理能力の低いユーザ側での計算量が増えてしまう．よって携帯電話といったような処理能力の低い端末での応用は望ましくない．

2.1.3 CINON 法

Lamport の方法におけるパスワードの再設定や処理量を解決する方式として CINON が提案された．この方式はサーバに対して以下の 3 つのデータを送信する．

1. 前回に正当性の確認が終わっている一方向性データの元のデータ
2. 次々回の認証に用いる一方向性データ
3. 前回送信されている一方向性データの正当性を確認するためのデータ

毎回，これらのデータを送信する事によって安全に認証情報の更新を行いつつ，認証を行っていく方式である．概要は以下に示す (図 2.2 参照)．

(1) 登録フェーズ ($n = 0$)

User: P，N_0，N_1 を任意に設定し，以下のデータを計算する．$E_0^0 = E(P, S \oplus N_0)$

$E_0^1 = E(P, E_0^0)$

$E_1^0 = E(P, S \oplus N_1)$

$E_1^1 = E(P, E_1^0)$

$M_0 = E(E_1^1, E_0^0)$

User: そして P，N_0，N_1 を保存する．

User: E_0^1，E_1^1，M_0 をサーバに送信する．

Host: E_0^1 を次回認証用データとして，

E_1^1 を次々回認証用データとして，(正当性の検証は次回に行う)

M_0 を E_1^1 の正当性を検証するために保存する．

(2) 認証フェーズ ($n = k$)

User: 記録してある N_{k-1} および N_k を読み出す．さらに，N_{k+1} を任意に設定し，以下のデータを計算する．

$E_{k-1}^0 = E(P, S \oplus N_{k-1})$

$E_n^0 = E(P, S \oplus N_k)$

$E_{k+1}^0 = E(P, S \oplus N_{k+1})$

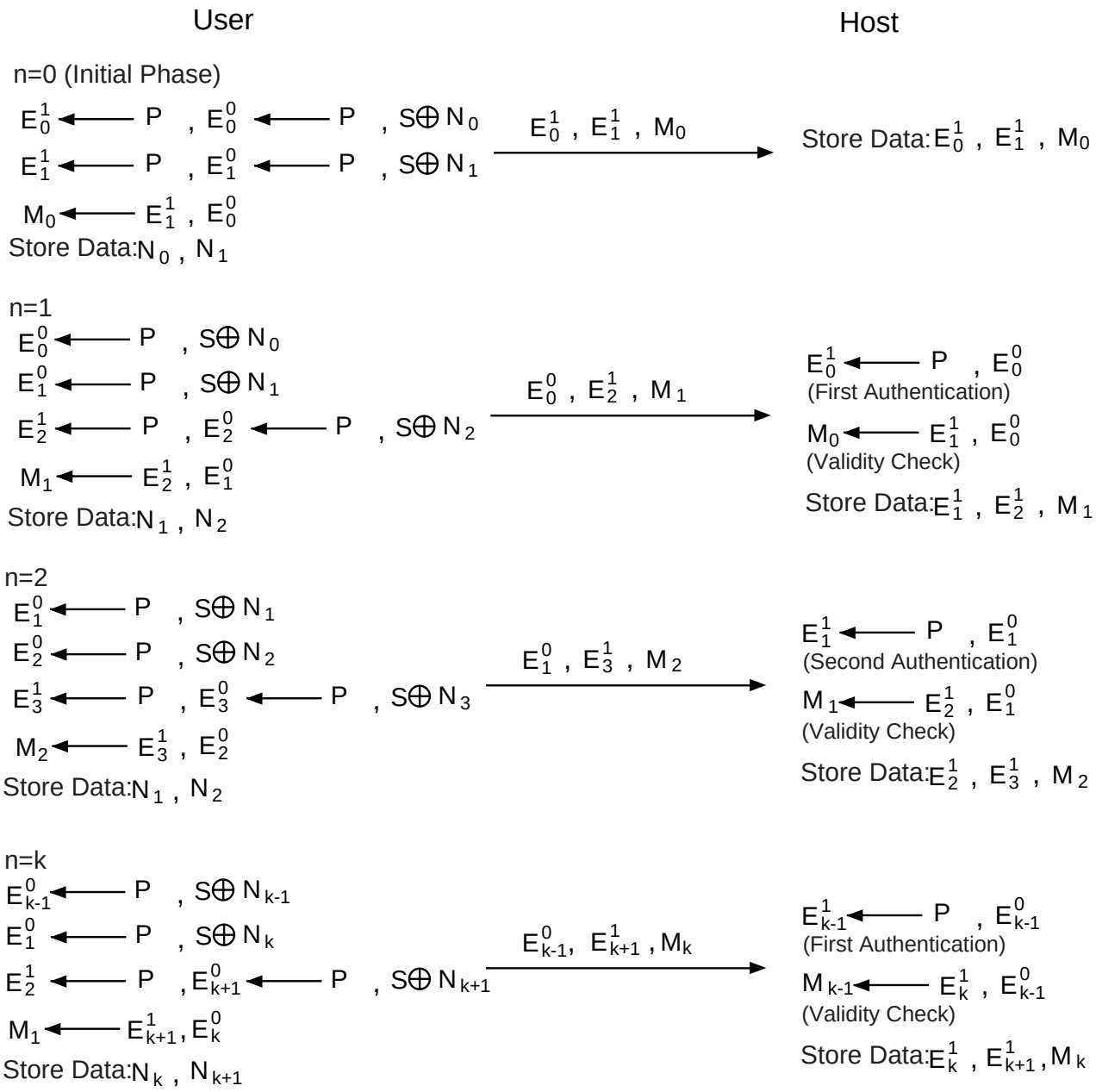

図 2.2 CINON 法の登録，および認証フェーズ

$$E_{k+1}^1 = E(P, E_{k+1}^0)$$

$$M_n = E(E_{k+1}^1, E_k^0)$$

User: E_{k-1}^0, E_{k+1}^1, M_k をサーバに送信する．

Host: 受信した E_{k-1}^0 から $E_{k-1}^1 = E(P, E_{k-1}^0)$ を計算する．

Host: 生成した E_{k-1}^1 と前回登録されている E_{k-1}^1 を比較，マッチすれば認証成功となる．

Host: さらに，E^0_{k-1} と記録してある E^1_k により，$M_{k-1} = E(E^1_k, E^0_{k-1})$ を計算し，前回記録してある M_{k-1} と比較．マッチしていたら先に受け取った E^1_k が正当であると判断する．

Host: E^1_k を次回認証用データとして，

E^1_{k+1} を次々回認証用データとして，(正当性の検証は次回に行う)

M_k を E^1_{k+1} の正当性を検証するために保存する．

CINON では理論的にはパスワードの再設定が必要とならず，永遠に認証が行える．しかし，攻撃者によって二回分の認証情報を立て続けに盗聴され，さかのぼって攻撃者にとって都合の良いデータにすり替えられる事により，なりすましの被害を受けるというセキュリティ上の問題がある．また Lamport に比べて処理量は減ったものの，送信データ量，記録データ量ともに Lamport の約 3 倍となっている．

2.1.4 SAS

Lamport や CINON といったプロトコルの問題を解決した方式として，SAS パスワード認証方式が提案されている．この方式はワンタイムパスワードの中で最も少ない計算量とプログラムサイズで，そして最も通信路上の盗聴による攻撃に強いプロトコルである．SAS は．前回の認証フェーズで登録しておいた一方向性関数を二回適用したデータの元のデータと次回認証用データを今回認証用データと次回認証用データに一方向性関数を適用したものでマスクして送り，認証と同時に次回認証用データの正当性の検証と登録を行うという方式である．概要は以下に示す (図 2.3 参照)．

(1) 登録フェーズ ($n = 0$)

User: $E^2_0 = E^2(S \oplus N_0)$ を計算する．

User: A, E^2_0 をサーバに安全なチャネルを用いて送信する．

Host: A, E^2_0 を保存する．

(2) 認証フェーズ ($n = k$)

User: 以下のデータを計算し，ユーザ ID A と共に Host に送信する．

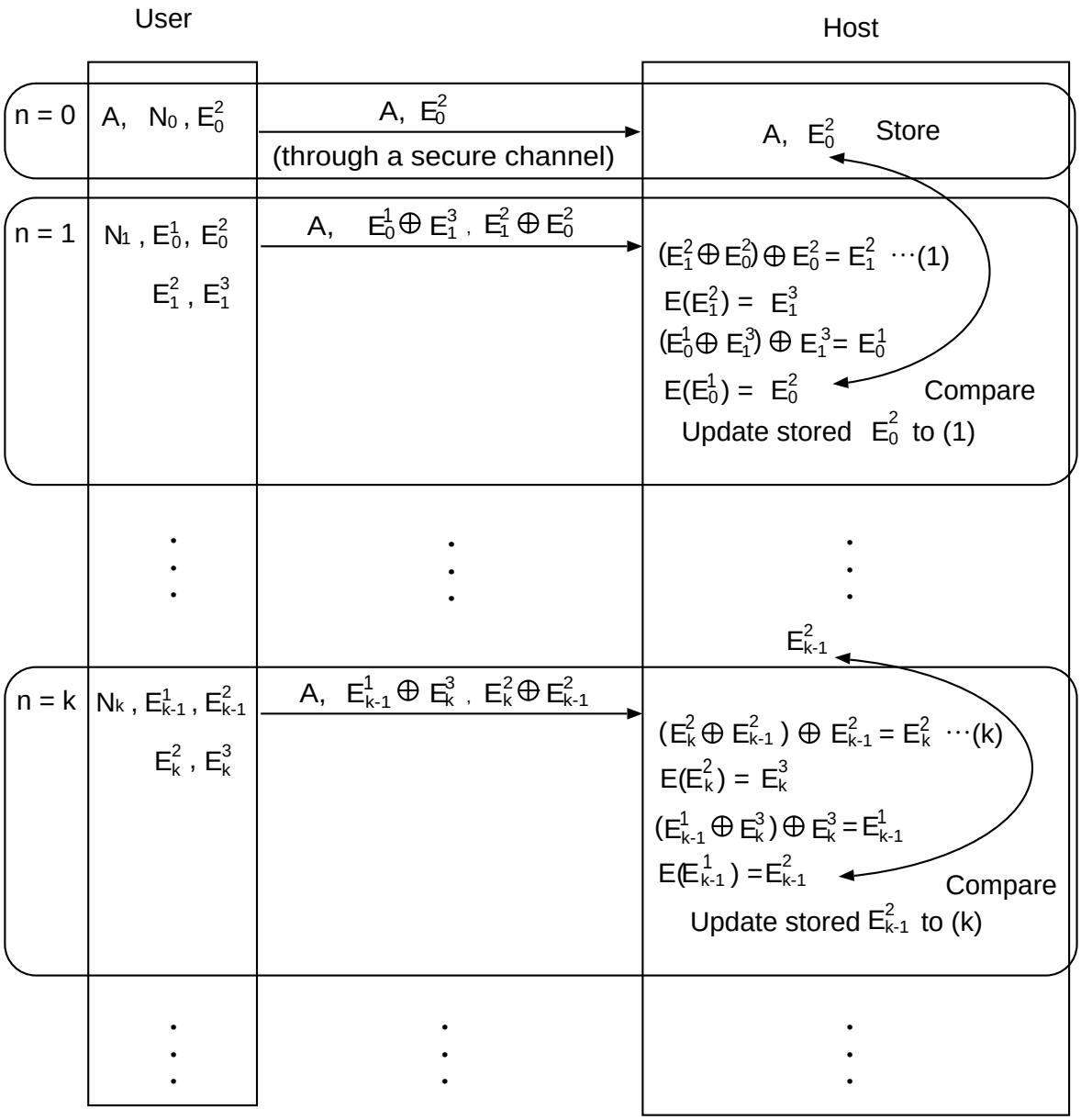

図 2.3 SAS の登録，および認証フェーズ

$E_k^1 \oplus E_k^3 + 1,\ E_{k+1}^2 \oplus E_k^2$

Host: $E_{k+1}^2 \oplus E_k^2$ と，保存されている E_k^2 との排他的論理和を取り，以下を取得する．

E_{k+1}^2

Host: E_{k+1}^2 にハッシュ関数を適用する．

$E(E_{k+1}^2) = E_{k+1}^3$

取得した E_{k+1}^3 を $E_k^1 \oplus E_{k+1}^3$ と排他的論理和を取り，今回認証用中間データ E_k^1 を取得する．E_k^1 にハッシュ関数を適用し，保存している E_k^2 と比較する．マッチすれば，User は認証され，次回認証フェーズのために，E_k^2 を E_{k+1}^2 に更新する．

アンマッチならば認証は拒否される．

SAS では認証を行う度に次回用鍵の登録を行うので Lamport と違い，パスワードの再設定の必要がない．また処理量や記憶量，データ通信量も図 2.4 のように少なくてすむ．安全性に関しても，認証情報の再利用攻撃やサービス不能攻撃に対して耐性がある．

Performance / *Method*	*Host*		*User*		*User-Host*	
	Hash Interation (time)	*Data Storage (bits)*	*Hash Interation (time)*	*Data Storage (bits)*	*Transmission Interation (time)*	*Transmission Bulk (bits)*
Lamport	*1*	*70*	*100-1000*	*64 for n*	*S.R(2 way) & 1(1 way)*	*70*
CINON	*2*	*200*	*5*	*128 for N1&N2*	*1*	*200*
SAS	*2*	*70*	*5*	*0*	*1*	*140*

図 2.4　Lamport，CINON，SAS の性能比較

2.2　SAS における同期問題

SAS は軽量かつ安全なプロトコルとして，計算能力の低い携帯電話などへの実装が検討されてきた．その実装の段階で問題となったので，同期問題である．同期問題とは，SAS プロトコルではユーザが認証フェーズ毎に生成した乱数を元に認証用データを作り出して認証を行う．しかし第三者からの攻撃や通信路断絶といったような何らかの理由でサーバだけ乱数が新しいものに更新されてしまい，次回の認証時にユーザの使用する乱数とサーバが使用する乱数にずれが発生して正常な認証が行えなくなる問題である (図 2.5 参照)．この同期問

題が発生しうる事象について，以下のように切り分けて考える事ができる．

1. ユーザからの認証情報がサーバに届かなかった場合
2. サーバでの認証処理が正常に終了しなかった場合
3. サーバでの認証処理終了後に，認証成功の応答がユーザに届かなかった場合
4. 悪意のある第三者により盗聴，なりすまし認証が行われてた場合

1 については，ユーザ，サーバともに認証情報の更新が行われないため，同期問題は生じない．また，2 については認証情報の更新の前であれば 1 と同様に考えられ，更新後であれば 3 と同様のと考えられる．3 と 4 ではサーバのみが新しい認証情報に書き換わってしまうために同期ずれが発生してしまう．

以上の事から，対策を考慮すべき点は 3 と 4 という事になる．

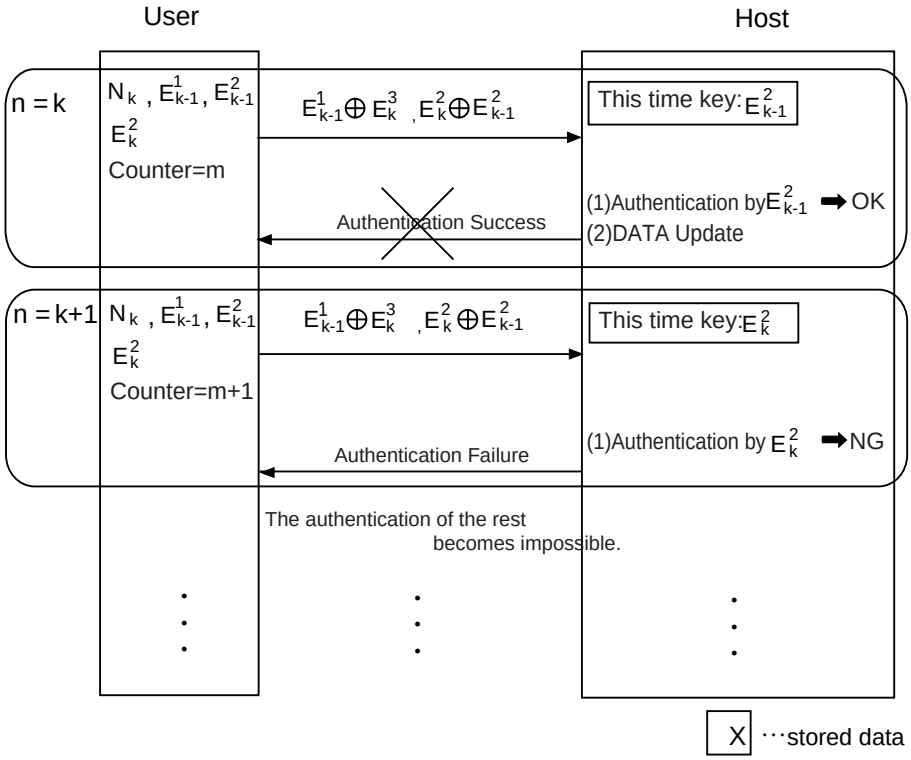

図 2.5　同期問題

2.3 フラグを用いる同期管理方式

SAS の同期管理方式として本学卒業生によりフラグを用いて同期管理を行う方式が提案されていた [5]．ここでは本学卒業生の大石 恭弘が提案した，サーバとユーザが交互に同じフラグを保持するという同期管理方式について説明する．

2.3.1 概要

通信が切断される場合には，ユーザからサーバへの認証用データが未達の時，サーバからユーザへの認証成功のメッセージが未達の時の 2 つが考えられる．これらの問題に対し，この方式ではユーザ，サーバ共にフラグ (0，もしくは 1 の値を取る) を使用して同期管理を行う．概要は図 2.8 で示す．

(1) 登録フェーズ

・ユーザとサーバのフラグをそろえておく．図 2.8 の場合では”0”である．

(2) 認証フェーズ ($n = k$)

User: 認証フェーズ毎にユーザ ID と認証用データに加え，フラグの値もサーバに送信する．

Host: フラグの値によって，同期が取れているかどうかをチェック．保持しているものと同じ値であれば，同期チェックは成功とし，認証成功するとサーバはフラグを”1”に更新し応答を返す．違っている場合は同期エラーのメッセージをユーザに送信してフラグの値を修正させる．

User: ユーザは認証成功のメッセージを受け取るとフラグの値を更新する．

このようにしてユーザ側とサーバ側で同じフラグの値”0”と”1”を交互に保持ようにする．もしこの方法において応答が途絶えた場合，次のフェーズでフラグがサーバ側でフラグが違っている事を検出し，ユーザに対して同期エラーメッセージを返す．ユーザは同期エラーメッセージを受け取り，前回の認証フェーズで認証成功したものの，応答を受信出来なかっ

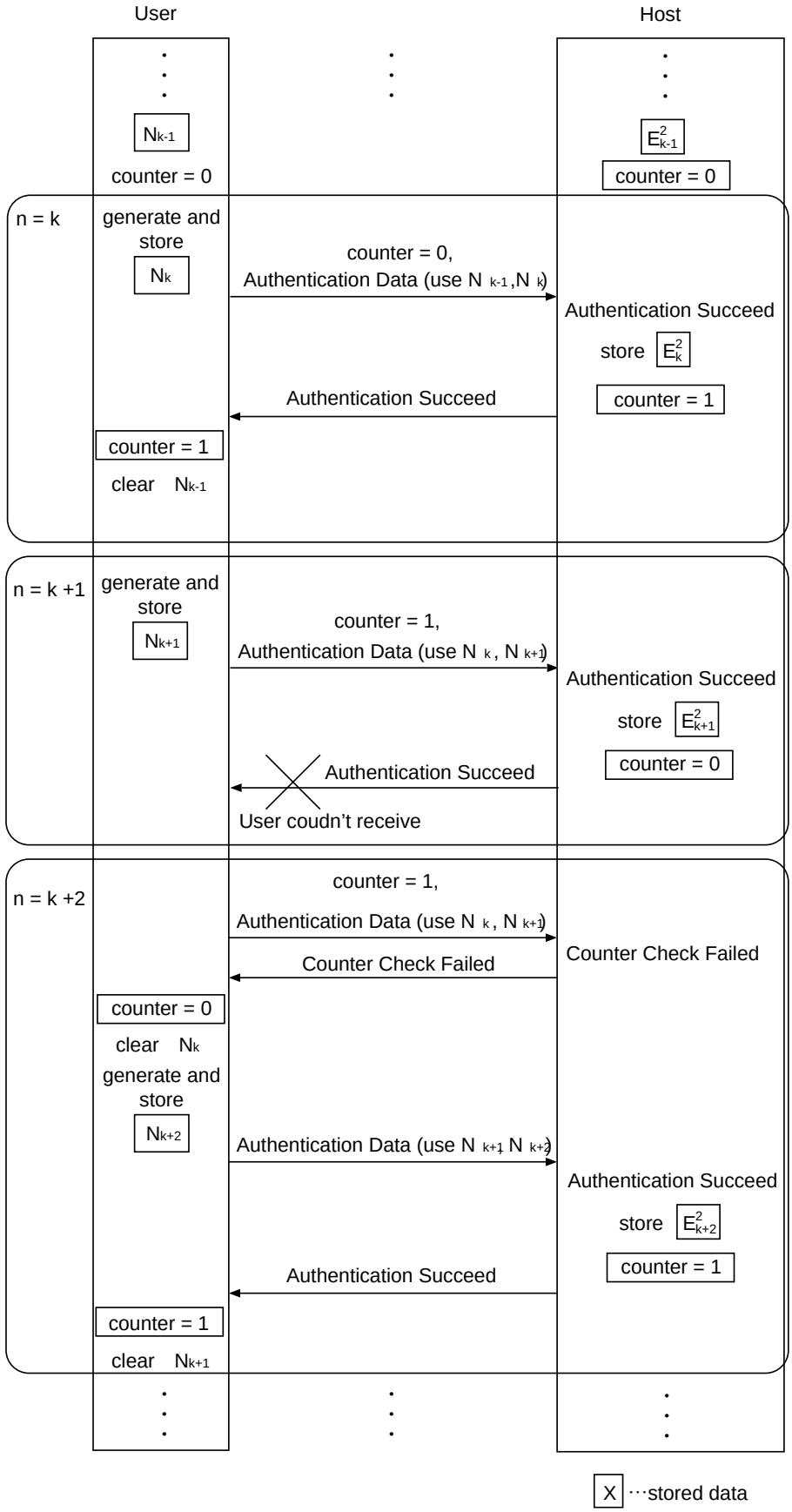

図 2.6　フラグを利用した同期管理方法

た事に気付く．そして次回用鍵を今回用鍵に置き換えて認証を行い，同期ずれの修正を行う事が出来る．

2.3.2 問題点及び攻撃法

この方式では第三者からのフラグの改ざんを想定していないので，フラグの値をそのまま送信している．つまり第三者によって容易に有意味な値に改ざんが可能である．またフラグの値のみで同期のずれをチェックしようとしているので，フラグの値を変更する事によって再送とサーバ側に勘違いさせる事が可能となる．

次に具体的にこの方式に対する攻撃法を解説する．図 2.7 を例にとって説明すると，まず，ユーザがサーバに対して正常なフラグと認証情報を送信した際に，攻撃者はフラグの値を変更する (図 2.7 の場合では”0”から”1”に変更)．サーバは送信されてきたフラグの値をチェックし，本来は同期ずれが起きていないのにも関わらず，フラグの値より同期ずれと判断して同期エラーメッセージをユーザに返してしまう．同期エラーメッセージを受け取ったユーザ側でも同期のずれが起こったと判断して，次回用鍵を今回用鍵に置き換え，新たに次回用鍵を作り直し，フラグの値を修正して送信する．攻撃者は再びフラグの値を改ざんする．これによってフラグの値は正常な状態に戻り，同期チェックは成功する．しかし，ユーザ側の今回用鍵だけが更新してしまう事によって同期のずれが生じ，認証自体が失敗に終わってしまい，以降の認証が不能となってしまう．

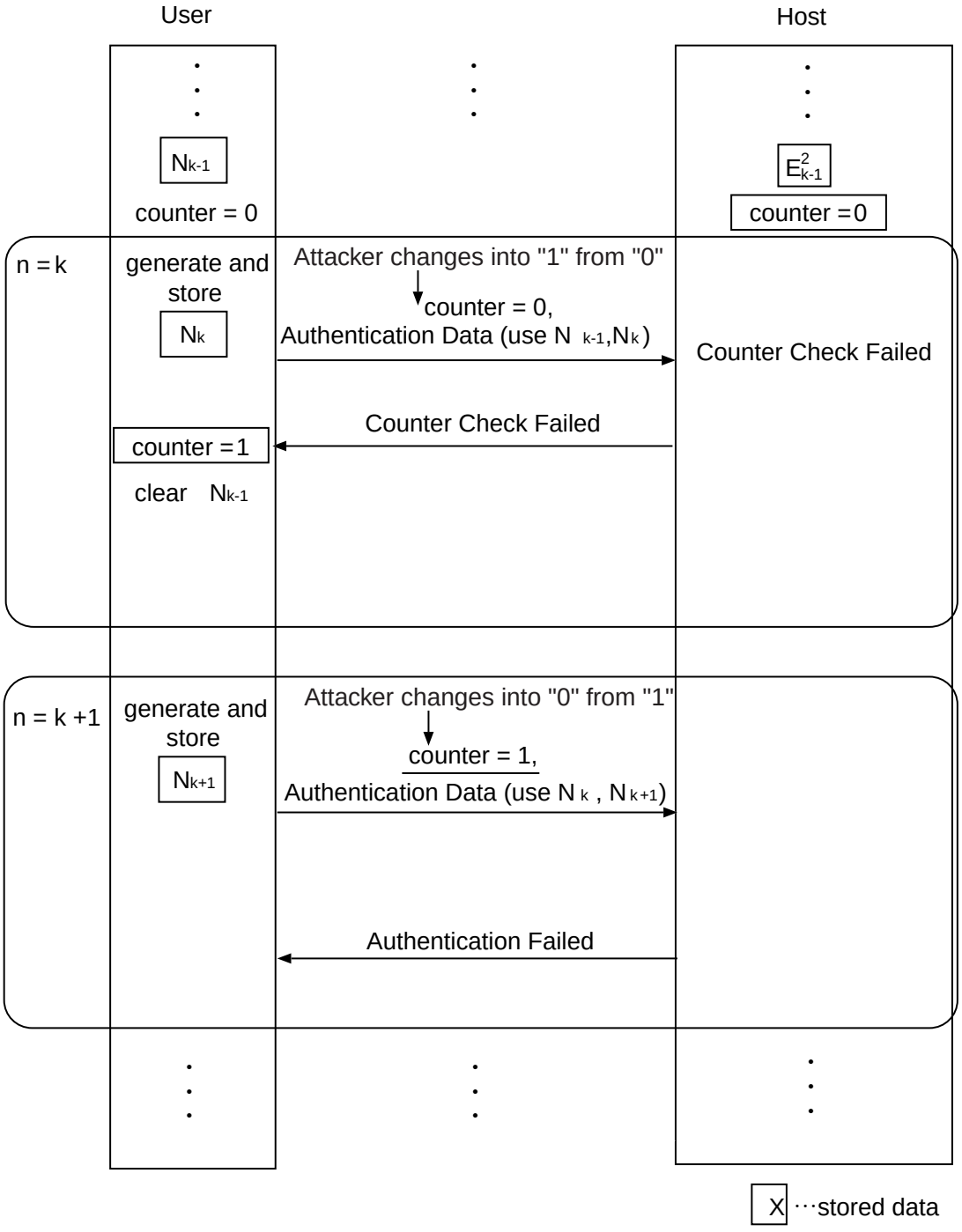

図 2.7 フラグのすり替え

2.4 カウンタを用いた同期管理方式

現在，SAS プロトコルは NTT アドバンステクノロジ株式会社により，携帯端末用個人認証システム”MobileSAFE”として商品化されている．この実装の際に，同期管理が大きな問題となった．そこで彼らはカウンタを用いた独自の同期管理方式を採用して，この問題の解決を図った [6]．以下に，そのカウンタ同期方式の概要と性能について説明する．

2.4.1 概要

この方式ではシーケンシャルナンバをカウンタとして使用し，今回認証用の鍵で暗号化して送り，カウンタの値が進んでいるかチェックする事によって再送のチェックを行う．概要は図 2.8 に示す．

(1) 登録フェーズ

・乱数でカウンタの初期値を決定し，ユーザが保持する．サーバのカウンタは”0”とする．

(2) 認証フェーズ $(n = k)$

User: カウンタを今回認証用の鍵と用いて暗号化し，認証情報と共にサーバに送信する．

Host: まず保持している今回認証用の鍵を用いて認証を行う．認証成功した場合は，今回認証用の鍵を用いてカウンタを復号し，保持しているカウンタと大小比較する．

Host: もし今回認証用の鍵で認証が失敗した場合は，再送であるか検査するために保存しておいた前回認証用の鍵を用いて認証を行う．認証が成功した場合は前回認証用の鍵を用いてカウンタの復号を行い，保持しているカウンタと大小比較する．

Host: 今回用もしくは前回用の鍵のどちらかで認証が成功し，カウンタのチェックが成功した場合は認証成功のメッセージをユーザに送信する．今回認証用と前回認証用の鍵の両方で認証失敗した場合は認証失敗のメッセージを送信する．またカウンタの値が不正であった場合も認証失敗とする．

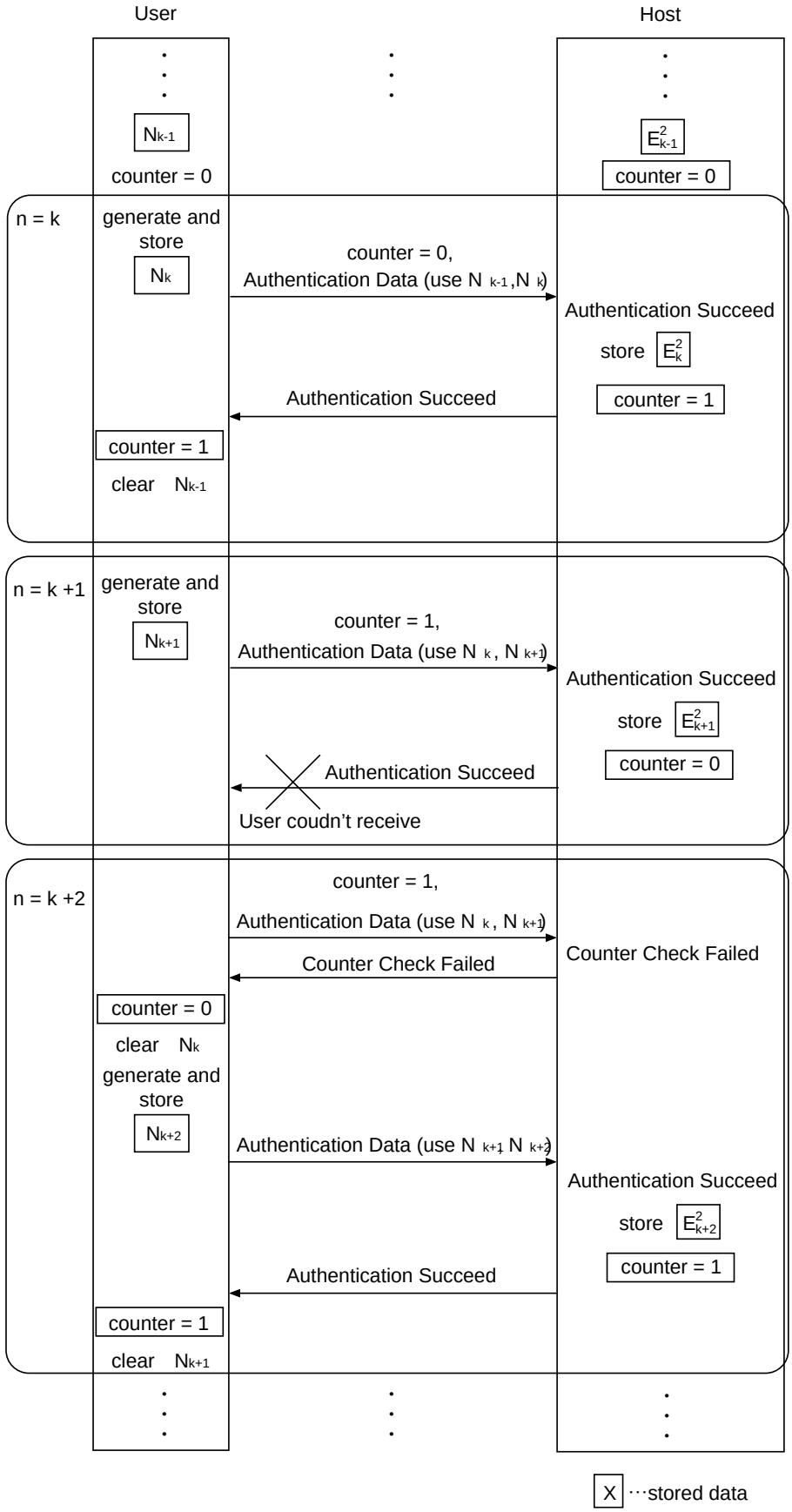

図 2.8　カウンタを利用した同期管理方法

2.4.2　性能

携帯電話での実装を考慮して，この方式ではユーザである携帯電話側での処理を減らす工夫がされている．まず一つ目は，ユーザが再送を行っていないかをチェックするために，サーバ側で認証が失敗した際に前回の鍵を使用して認証処理を行っている．これによって処理能力の低いユーザ側での計算を減らす事が出来る．そして次に，この方式が第三者が認証情報を盗聴し，再送を装って認証を行おうとする事を防ぐために，カウンタを今回認証用の鍵で認証毎に新たに暗号化して送っている．これによってカウンタまで同じものは正規ユーザから一度も送られないので，第三者からのアクセスを防止する事ができる．また，このカウンタは常にインクリメントされるだけで，初期化されない．初期値については乱数発生させているので第三者が通信回数を数えていたとしても容易にはカウンタの値を予想できない．カウンタの値は java の lang でとっているため，取り得る値は -2^{63} から $2^{63}-1$ となっており，有限であるが実用上は問題ない程度に多い．もし正規ユーザ以外がなりすまし認証を成功させたとしても，ユーザはカウンタをインクリメントしているので以降の認証に問題は起きないようになっている (図 2.9 参照).

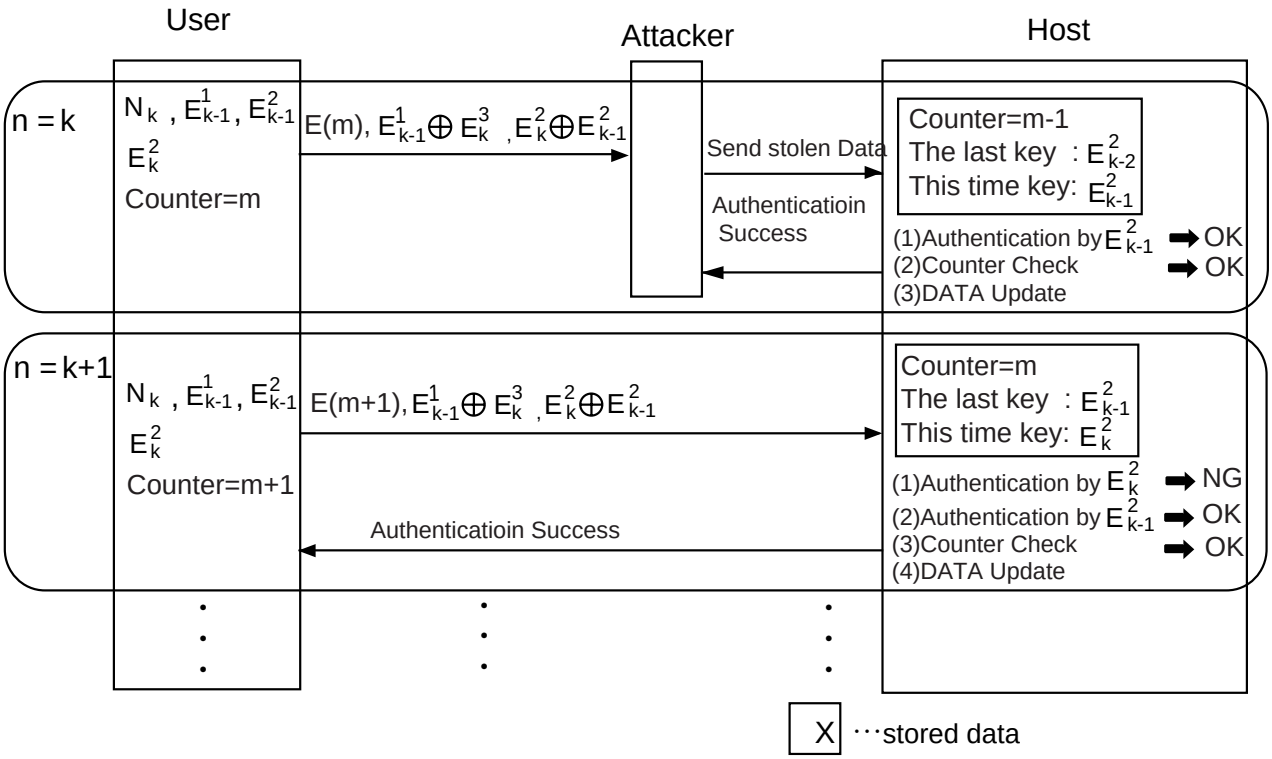

図 2.9　なりすましが行われた場合

第 3 章

カウンタ同期方式の問題と攻撃法

3.1 カウンタ同期方式の問題点

カウンタ同期方式では今回認証用の鍵を用いてカウンタを暗号化する．これによって今回認証用の鍵を含んでいる認証情報を保障する事ができるが，これでは次回認証用の鍵が本当にユーザの意図したものかは保障できない．今回認証用の鍵は同じで，次回認証用の鍵だけ違う認証情報をすり替えて送る事が出来たら，正規ユーザは以降の認証ができない状態になる．この問題点に着目して私は盗聴，すり替えによるサービス不能攻撃を発見した．

3.2 カウンタ同期方式に対するサービス不能攻撃

カウンタ同期方式ではユーザは今回認証用の鍵を使用してカウンタを暗号化し，サーバ側では認証情報から今回認証用の鍵を取得し，カウンタを復号化する．一見はこれによってカウンタと認証情報に関連性が生まれ，認証情報やカウンタのすり替えが行えないように思われる．しかし先ほど述べたように，次回認証用の鍵だけ違う認証情報をすり替えて送る事が出来たら，正規ユーザは以降の認証ができない状態になる．そこで私は認証失敗した際に，ユーザが次回認証用の鍵を作り直し，新たに今回認証用の鍵は同じで，次回認証用の鍵が違う認証情報を作成する点に注目した．それでは発見した攻撃法を説明する．

図 3.1 の例を使用して説明する．k 回目の認証フェーズにおいて，攻撃者は正規ユーザからの認証情報”$E^1_{k-1} \oplus E^3_k$, $E^2_k \oplus E^2_{k-1}$”を盗聴し，認証情報を同じ長さの乱数に

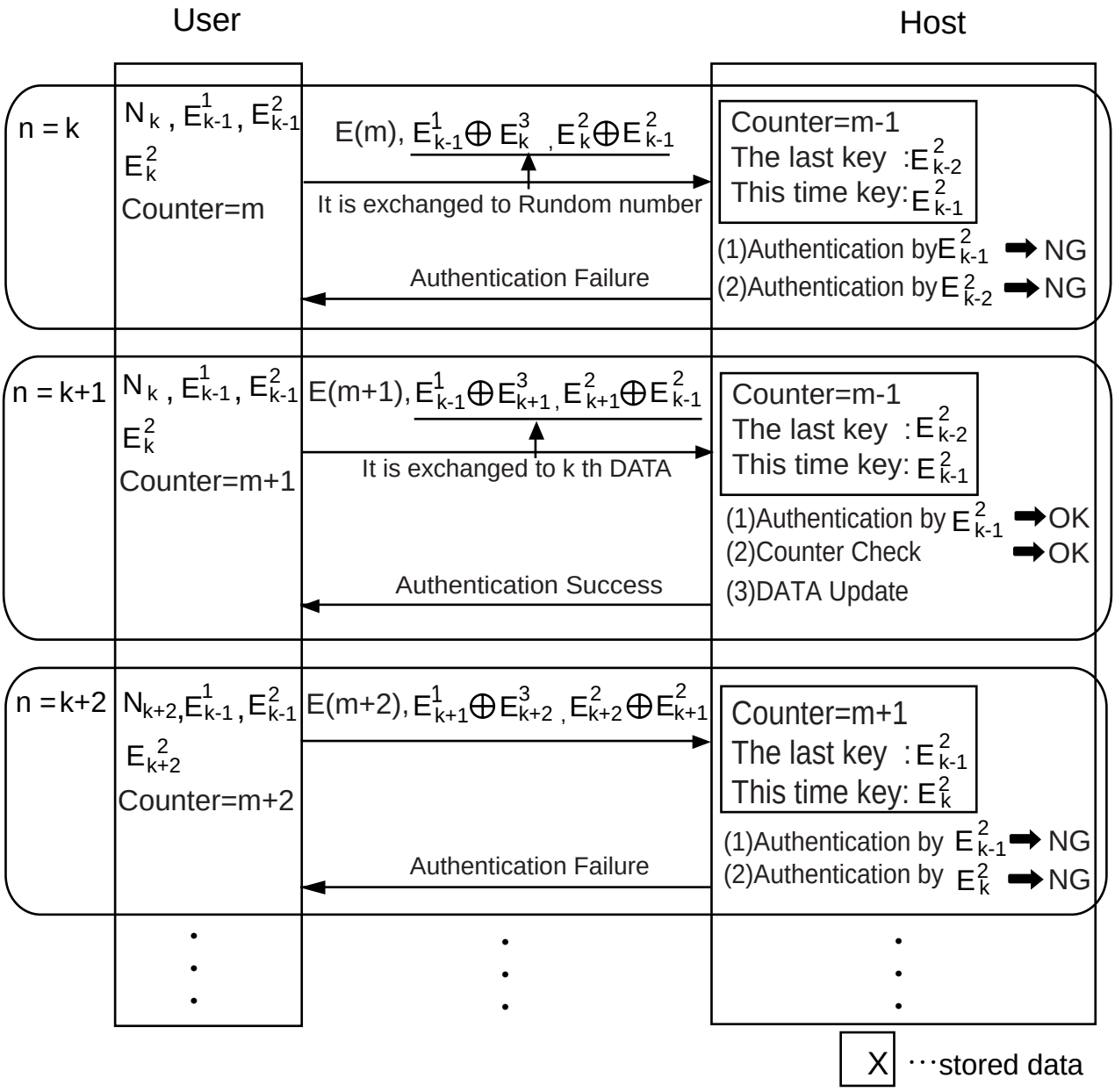

図 3.1　カウンタ同期方式に対するサービス不能攻撃

すり替えて送信する．サーバは受け取った認証情報で認証を行うが，攻撃者によって適当な値に変更されているので，もちろん認証失敗の応答をユーザに送信する．$k+1$ 回目の認証フェーズに認証失敗の応答を受け取った正規ユーザはパスワードの入力が失敗したと判断して，新たにパスワードの入力を行い，前回と同じ乱数"N_{k-1}"を使用して今回認証用の鍵を作成する．しかしここで次回認証用の鍵"E^2_{k+1}" を新しい乱数"N_{k+1}"を使用して作り出し

てしまう．こうして正規ユーザは先程と今回認証用の鍵が同じで次回認証用の鍵だけが違う認証情報”$E^1_{k-1} \oplus E^3_{k+1}$, $E^2_{k+1} \oplus E^2_{k-1}$” を作成して $k+1$ 回目の認証フェーズに送信する．そこで攻撃者は前回，正規ユーザが再送した時の認証情報”$E^1_{k-1} \oplus E^3_k$, $E^2_k \oplus E^2_{k-1}$”と今回の認証情報をすり替えて送信する．これによって認証自体は成功するが，次回認証用の鍵が正規ユーザが意図していた”E^2_{k+1}”ではなく，”E^2_k”が登録されてしまう．この $k+2$ 回目の認証は成功するが，次回の $k+3$ 回目以降の認証はユーザの使用している認証情報の乱数とサーバ側で保持している認証情報の乱数が異なるので認証できなくなる．

第 4 章

解決法の提案と評価

4.1 解決案

この攻撃法はカウンタ同期方式がカウンタを暗号化した鍵と，認証情報の今回認証用の鍵が同じであれば正規ユーザからの認証であると判断するという所が弱点となっている．つまり既存方式の問題点はカウンタと認証情報との関連性がない事が問題点となっている．そこで私はカウンタの暗号化の鍵に今回認証用の鍵と次回認証用の鍵にハッシュ関数を一度適用したものを排他的論理和演算したものを使用する方法を提案する。これによって暗号化されたカウンタと認証情報に関係を持たせる事が出来る．

4.2 評価

4.2.1 安全性

カウンタ同期方式において，問題であったカウンタと認証情報との関連性がない点であった．本方式はカウンタの暗号化の鍵に今回認証用の鍵だけでなく，次回認証用の鍵に関連したものを使う事によって問題を解決する事に成功した．この方式によって何故，カウンタと認証情報に関連性を持たせる事が出来たと言えるを説明する．SAS プロトコルの認証情報はユーザ ID 以外に二つある．一つは今回認証用の中間データと次回認証用の鍵に一回ハッシュ関数を適用したものとを排他的論理和演算したもの．もう一つは次回認証用の鍵と今回認証用の鍵を排他的論理和演算したの二つである (図 4.1 参照)．

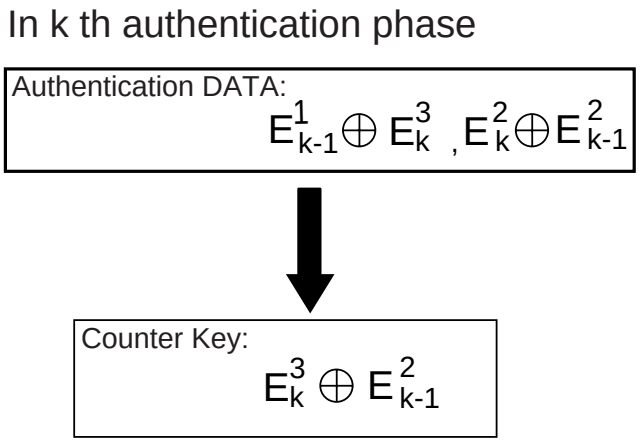

図 4.1 提案方式におけるカウンタの鍵

既存方式では今回の鍵を使用して二つ目のデータに含まれる，今回認証用の鍵を使用し今回の鍵の検証を行うと共に，SAS プロトコル自身の次回認証用の鍵を検証するメカニズムによって認証情報を保障しているように考えられるが，SAS プロトコルの次回認証用鍵の正当性検証はあくまでも，一つ目の認証情報と二つ目の認証情報の間にある次回認証用の鍵が同じものであるかを検証しているのであって，カウンタが作られた時の次回認証用の鍵という保証にはならない。カウンタと認証情報の組み合わせに関連性を持たせるためには，一つ目の認証情報と二つ目の認証情報のそれぞれから，今回認証用の鍵と次回認証用の鍵に関係したものを取り出さなければならない．そこで今回の提案方式では一つ目のデータに含まれる，次回認証用の鍵に一回ハッシュ関数を適用したものと，二つ目の式に含まれる，今回認証用の鍵を使用する事によって，この二つのデータにカウンタを関連付けた．これによって今回認証用の鍵のみなず，次回認証用の鍵をも保証する事が可能となる．

再利用攻撃

もし攻撃者が認証情報を改ざんし，なりすまし等の攻撃を行おうとするならば，まずカウンタの鍵を作成し，取得したカウンタを新たに改ざんした認証情情報を使用して作成する必要がある．そこでこの提案方式で使用されているカウンタの鍵が通信路上の盗聴によって作成される危険性がないかの検証を行う．

ここで第三者が，なりすましを成功させるために必要なデータについて考えてみる (図 4.2 参照).

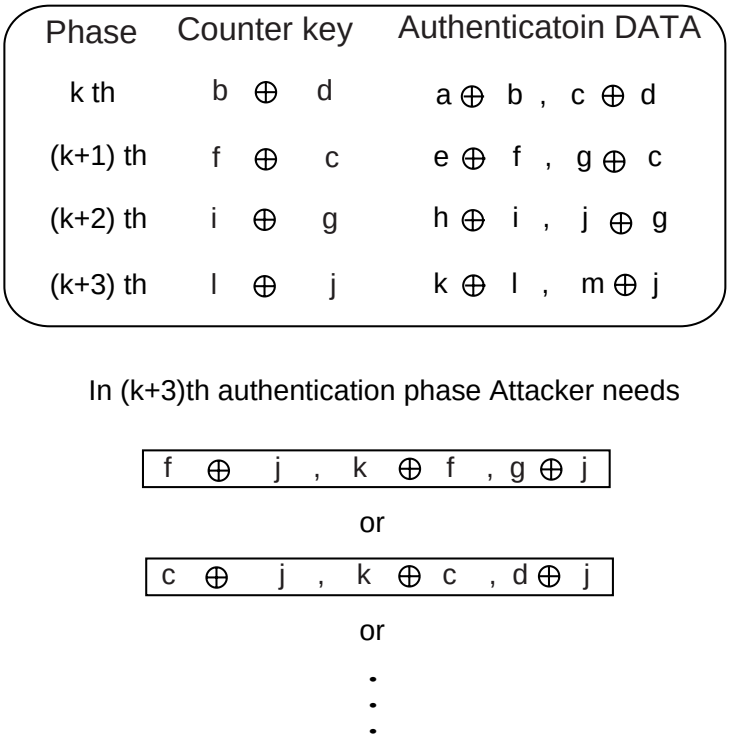

図 4.2 提案方式に対する再利用攻撃

第 k 回目の認証フェーズにおいて，User から送信される認証情報

$E_{k-1}^1 \oplus E_k^3$, $E_k^2 \oplus E_{k-1}^2$

とカウンタの暗号化に使用する鍵

$E_k^3 \oplus E_{k-1}^2$ は，

認証情報:$a \oplus b$, $c \oplus d$, カウンタの鍵：$b \oplus d$

と表す事ができる．同様に第 $(k+1)$ 回目から $(k+3)$ 回目の認証情報をそれぞれ

カウンタの鍵：$f \oplus c$ ，認証情報:$e \oplus f$, $g \oplus d$…第 $(k+1)$ 回目

カウンタの鍵：$i \oplus g$ ，認証情報:$h \oplus i$, $j \oplus g$…第 $(k+2)$ 回目

カウンタの鍵：$l \oplus j$ ，認証情報:$k \oplus l$, $m \oplus j$…第 $(k+3)$ 回目

と表せる．ここで，第三者によってこれまでの全ての通信を盗聴したとし，例えば第 (k+3)

目の認証フェーズでユーザから送られたカウンタを盗聴し，認証情報から生成した鍵を使用して復号しようとした場合，"$l \oplus j$" が必要になる．このデータの"l"を作成する場合，第$(k+3)$回目の認証情報である"$k \oplus l$"から取り出す事になる．しかし，"$k \oplus l$"から取り出すためには"k"で排他的論理和演算して取り出す必要があるが，kは"$k \oplus l$"にしか出てこない．排他的論理和は同じもの同士は打ち消してしますので，もし"$(k \oplus l) \oplus (k \oplus l)$"と演算した場合，答えは0となってしまう．この次回認証用の鍵にハッシュ関数を一度適用したものが複製できないので，鍵自体の複製も出来ないという事になる．

サービス不能攻撃

カウンタ同期方式で問題であったサービス不能攻撃について，今回提案した方式で考えてみる．

図4.3の例を使用して説明する．k回目の認証フェーズにおいて，攻撃者は正規ユーザからの認証情報"$E^1_{k-1} \oplus E^3_k,\ E^2_k \oplus E^2_{k-1}$"を盗聴し，認証情報を同じ長さの乱数にすり替えて送信する．サーバは受け取った認証情報で認証を行うが，攻撃者によって適当な値に変更されているので，もちろん認証失敗の応答をユーザに送信する．$k+1$回目の認証フェーズに認証失敗の応答を受け取った正規ユーザはパスワードの入力が失敗したと判断して，新たにパスワードの入力を行い，前回と同じ乱数"N_{k-1}"を使用して今回認証用の鍵を作成する．そしてここで次回認証用の鍵"E^2_{k+1}" を新しい乱数"N_{k+1}"を使用して作り出す．そして正規ユーザは先程と今回認証用の鍵が同じで次回認証用の鍵だけが違う認証情報"$E^1_{k-1} \oplus E^3_{k+1},\ E^2_{k+1} \oplus E^2_{k-1}$" と，前回と同じ今回用鍵と新たに生成した次回用鍵にハッシュ関数を一度適用したもので作った"$E^3_{k+1} \oplus E^2_{k-1}$"をインクリメントしたカウンタの鍵として暗号化したものを$k+1$回目の認証フェーズに送信する．そこで攻撃者は前回，正規ユーザが再送した時の認証情報"$E^1_{k-1} \oplus E^3_k,\ E^2_k \oplus E^2_{k-1}$"と今回の認証情報をすり替えて送信する．サーバはまず，今回用の鍵で認証し，認証成功する．カウンタ同期方式ではこの後，カウンタのチェックも成功し認証成功となるが，提案方式のカウンタには次回用鍵に関係し

4.2 評価

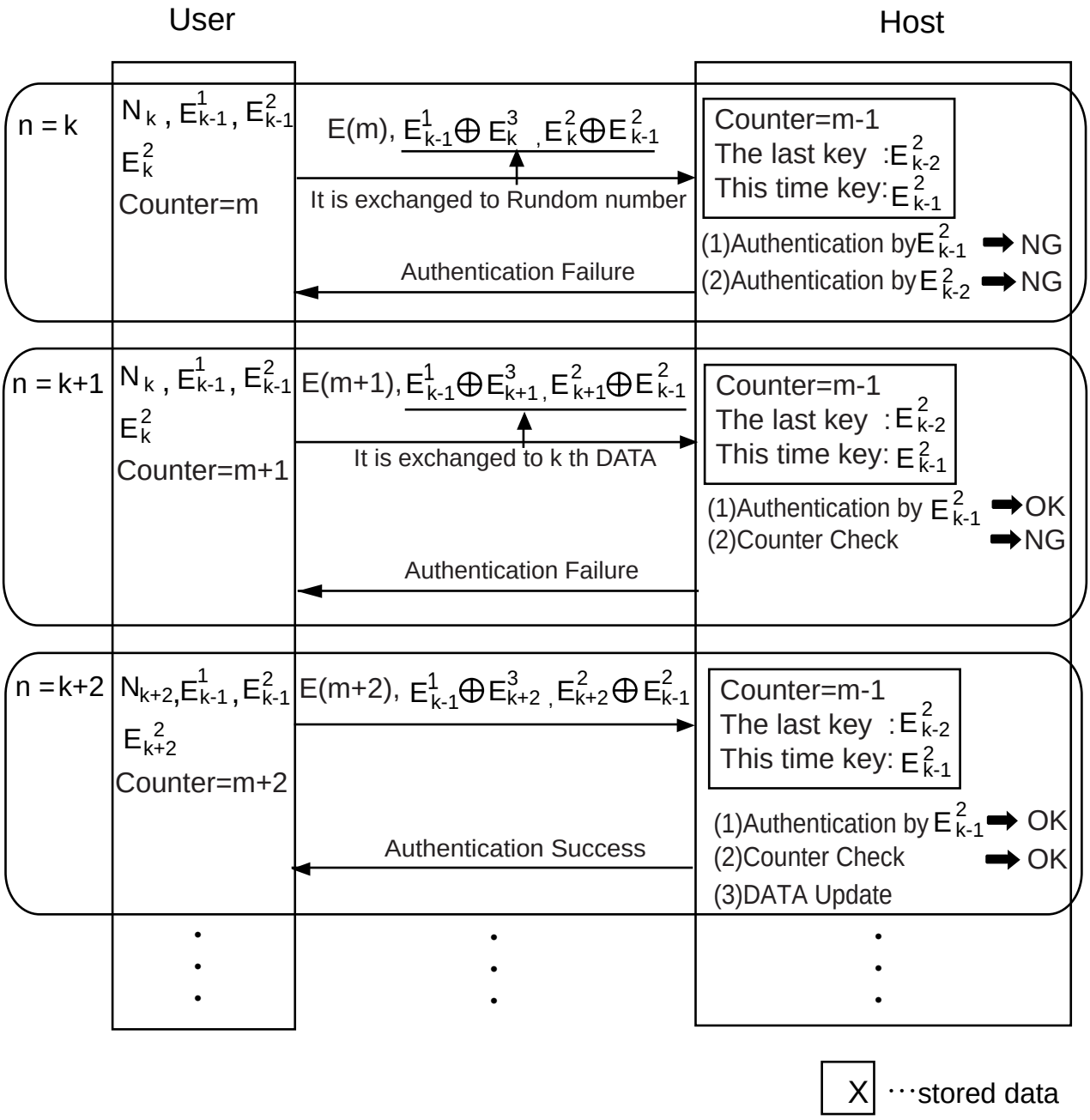

図 4.3 提案方式に対するサービス不能攻撃

たものを使用しているのでカウンタの復号の時点で失敗してしまい，第三者による改ざんを検出できる．

4.2.2 処理量

計算量であるが，今回認証用の鍵と認証の際に得られた次回認証用の鍵にハッシュ関数を一度適用したものと排他的論理和演算する必要があるので排他的論理和演算が以前より一つ増える事になる．しかし他の処理は全くカウンタ同期方式と同じで出来るので，カウンタ同期方式のシステムを大幅に変更する必要はなく問題を解決する事が出来る．

第 5 章

おわりに

インタネットおよびモバイルコミュニケーション環境において，ユーザや通信相手の資格認証が必要不可欠なものとなっている．資格認証の方法の中には SAS ワンタイムパスワード認証方式という極めて簡易かつ安全なプロトコルが提案されており，同期管理という実装上の問題を解決し，携帯端末用個人認証システム”MobileSAFE”として商品化されている．本論文では，SAS を実装した際に用いたカウンタ同期方式が今回認証情報の検証だけを行い，次回認証情報の検証を行わないで認証成功させるという問題があり，その問題を利用し，次回認証情報だけが違う認証情報をすり替えられて，サービスが不能になるという問題がある事を指摘した．そしてこの次回認証情報を検証しない事による認証情報のすり替えを回避するために，カウンタの鍵に今回用鍵だけでなく，次回用鍵に関係したものを使用する事によってカウンタの復号の際に次回認証情報の検証も行えるという解決法の提案を行った．これによってすり替えを防ぐ事が出来るようになっただけではなく，認証情報とカウンタとの間に一意な関係を持たせる事によって，カウンタか認証情報のどちらかだけを改ざんする事が出来ないようにし，改ざんの検出が行えるようになった．この提案法によって増加した処理は排他的論理和演算一つだけと極めて少ない追加コストで問題の解決が可能となった．

今後の課題であるが，現在の方式では今回用鍵で認証が失敗する度に，認証情報が再送されてたものでないかをチェックするために前回の鍵を用いて認証を行わなければならない．これではただ認証が単純に失敗した際にも前回の鍵を用いて認証情報が再送でないかのチェックを行わなければならない．そこで今後の課題として認証情報の再送をより簡易に行える方法の検討を行う必要がある．また現在の方式ではサーバの応答は単純のメッセージで

あるので，サーバの応答を偽るといったようなサーバのなりすましという危険性がある．そこでこの問題にプロトコルレベルで対応するために，ユーザの認証だけでなくサーバの認証も行える，相互認証も検討する必要もある．また現在の方式ではカウンタを用いる事によって同期の管理を行っているが，この方式では第三者からの改ざんを防ぐためにカウンタを暗号化をする必要があり，処理量が増えてしまう．今後の課題としてカウンタを用いない方式の検討も必要である．

謝辞

高知工科大学工学部情報システム工学科 清水明宏教授には，卒業研究を含め，多岐に渡って懇切に御指導，御教示を賜った．ここに深謝申し上げる．

また，本学大学院工学研究科基盤工学専攻 岡田 実氏，辻 貴介氏には研究途上において有益な御議論，御助言を頂いた．ここに心からお礼を申し上げる．

清水研究室学部生 中務秀和氏，河村智氏をはじめ研究室の方々には，研究途上において有益な御議論を頂いた．諸氏に心より感謝する．

参考文献

[1] L. Lamport, “Password authentication with insecure communications,” Commun.ACM, vol. 24, no. 11, pp. 770-772, 1981.

[2] A. Shimizu, “A Dynamic Password Authentication Method Using a One-way Function,” IEICE Trans., vol. J73-D-I, no. 7, pp. 630-636, 1981.

[3] A. Shimizu, T. Horioka, and H. Inagaki, “A password authentication method for contents communication on the internet,” IEICE Trans. Commun., vol. E81-B, no. 8, pp. 1666-1673, Aug. 1998.

[4] 上岡 隆, 清水 明宏, “ワンタイムパスワード認証方式 SAS の安全性に関する検討,” 電子情報通信学会技術研究報告書, OFS2001-48, No. 435, pp. 53-58, 2001.

[5] 大石 恭裕,“ モバイルコンテンツ課金プロトコルの検討 ”, 高知工科大学卒業論文 2001.

[6] 真島 大介, 羽田 知良, 渋谷 充喜, 清水 明宏, “モバイル端末への SAS（Simple and Secure）認証実装方式の検討,” 電子情報通信学会技術研究報告書, OFS2001-49, No. 435, pp. 59-63, 2001.