State Based Key Hop (SBKH) Protocol

Srinivasan, K.; Mitchell, S.

Proceedings of the Sixteenth International Conference on WirelessCommunications Wireless 2004, 2004

National Research Council Canada
Conseil national de recherches Canada

Institute for Information Technology
Institut de technologie de l'information

NRC · CNRC

*State Based Key Hop (SBKH) Protocol **

Srinivasan, K., and Mitchell, S.
July 2004

* published in the Proceedings of the Sixteenth International Conference on Wireless Communications Wireless 2004. Calgary, Alberta, Canada. July 12-14, 2004. NRC 47462.

Canada

State Based Key Hop Protocol

Kannan Srinivasan and Stephen Michell
National Research Council
Sydney, Nova Scotia, Canada
Email: {kannan.srinivasan, stephen.michell}@nrc-cnrc.gc.ca

Abstract

Sate Based Key Hop (SBKH) protocol is created to provide a strong, lightweight encryption scheme for battery operated 802.11 devices, such as the sensors in a wireless sensor network. Due to its simplicity and ease of maintenance compared to wireless-fidelity (Wi-Fi) protected access (WPA) version 1.0 with robust security network association (RSNA) key management, SBKH also targets small office home office (SOHO) users. SBKH will support ad hoc networking, power save (PS) mode and even handoffs. This protocol eliminates all the issues with wired equivalent privacy (WEP) protocol, using existing hardware and software as much as possible.

1 Introduction

1.1 General

Several flaws within wired equivalent privacy (WEP) have been identified making WEP easy to be cracked in minutes [SIR 2001]. 802.11 task group i (802.11 TGi) [Draft 2003] is working on a more robust security standard (802.11i) for both legacy and future 802.11 devices. The Wi-Fi alliance, an alliance formed by many 802.11 manufacturers, has taken parts of the 802.11 TGi proposals and developed a solution called Wi-Fi protected access (WPA). WPA version 1.0 (WPA 1.0) has the firmware upgrade implementation for legacy devices and WPA version 2.0 (WPA 2.0) will be targeting future devices.

Although WPA 1.0 with 802.1x and extensible authentication protocol (EAP) solves the major issues identified with WEP, it may introduce performance degradation in existing 802.11 devices [Walker 2002]. It is also very costly in terms of computation and power, and so is not suitable for battery-operated devices such as sensors.

WPA 1.0 under the pre shared key (PSK) mode is not based on EAP and is being used in many SOHO networks. It has significant security issues as identified in [Moskowitz 2003] and is also computationally expensive.

We propose an alternative security solution for IEEE 802.11 using RC4, but not limited to 802.11 and can also be applied and extended to other existing and future wireless protocols. As an alternate solution for 802.11, this proposal can eliminate the security problems associated with WEP without the expense and performance penalties of WPA. Our proposal, State Based Key Hopping (SBKH) protocol uses RC4 encryption scheme in a novel way that is more efficient than WPA.

SBKH solves all the major issues identified with WEP and WPA 1.0 with PSK. SBKH is computationally very cheap compared to WEP and WPA (1.0 and 2.0), reducing the power consumption for encryption and decryption. SBKH is suitable for battery-operated 802.11 devices and for SOHO users for its robustness and simplicity. SBKH also provides some level of security even among insiders that is not available within WPA 1.0 with PSK or WEP.

1.2 Background

1.2.1 Wired Equivalent Privacy (WEP)

802.11 defined an encryption scheme called wired equivalent privacy (WEP), to provide security to the 802.11 users. WEP is a symmetric encryption scheme in which a WEP key is known or shared between two communicating nodes. WEP uses RC4 algorithm to do per packet encryption. RC4 algorithm is a stream cipher scheme [FMS 2001, FM 2000, Mantin 2001] in which the data is encrypted by XORing data with the cipher stream generated by a RC4 seed. WEP uses the concatenation of WEP key (40 or 104 bits long) and the initialization vector (IV) (24 bits long), as the RC4 seed. For every new RC4 seed, RC4 reinitializes its states using key-scheduling algorithm (RC4-KSA). After reinitialization of its encryption states, RC4 generates the cipher stream using pseudo random generation algorithm (RC4-PRGA). Since the IV is sent in every packet, in clear, WEP carries out RC4-KSA and RC4-PRGA for every packet.

1.2.2 Flaws in WEP

It became apparent as IEEE 802.11 was being standardized that WEP was not going to provide the wired equivalent privacy desired. The following specific defects within WEP were noted:

IV Collision: Since WEP uses concatenation of WEP key and IV as RC4 seed, the generated cipher stream will be the same for same IVs if the WEP key does not change. This means that if an attacker can predict the unencrypted data (plain text) for a given IV, then the attacker could determine what the corresponding cipher stream is. Once the same IV is repeated, the attacker could make use of the previously identified cipher stream to predict the plain text, corresponding to the repeated IV.

Weak Key: Due to the nature of RC4-KSA and RC4-PRGA algorithms used within RC4, there exist some weak keys, in which specific pattern in first few octets of the RC4 seed will result in a corresponding pattern in the first few octets of the RC4 cipher stream [Walker 2002]. This property can be used to derive the RC4 seed and hence the WEP key after monitoring the IVs and their corresponding cipher streams.

Bit Flipping Attack: WEP uses a CRC-32 algorithm to compute integrity check value (ICV) on the plain text data. This ICV is concatenated with plaintext data and then is encrypted with cipher stream. Known weaknesses in CRC-32 algorithm make it easy to modify data and perform corresponding changes to ICV such that this modification gets unnoticed. An intruder can perform such modifications to previously sent packets and send them again. The receiving node will accept such modified packet as ICV will succeed and will forward that packet to the final destination, permitting the attacker to interfere with higher-level protocols or generate and analyze known error responses from the destination.

Forgery Attack: WEP has no protection against redirection of encrypted packets. This can happen when receiver address (RA) or destination address (DA) field within each frame is changed. Changing one of these fields will result in delivery of the frame to a wrong destination. Changing source address (SA) or transmitter address (TA) field is also possible and may result in redirection of the responses.

Replay Attack: Any intruder that intercepts any WEP encrypted frames can resend the same frame later. The access point will forward such frames, as frame replays succeed decryption.

1.2.3 Wi-fi Protected Access (WPA)

In order to repair the defects identified above, IEEE 802.11i task group developed two proposals [Draft 2003] for security for 802.11 devices: one for 802.11 legacy devices and the other for future 802.11 devices. The former encapsulates WEP functionalities by temporal key integrity protocol (TKIP). The latter works with a different encryption scheme called advanced encryption standard (AES) and requires change in hardware. WPA 1.0 is a subset of the former and WPA 2.0 is a subset of the latter.

WPA 1.0 solves IV collision by using extended IV referred to as TKIP sequence counter (TSC). TSC is of 48 bits instead of 24-bit IV. It avoids any replay attacks by rejecting frames with out-of-sequence TSC counter value. WPA also uses a not fully secure Michael as message integrity code (MIC) algorithm to protect frames from any modifications. MIC is performed over the MAC header and the plain-text data, and adds an eight-byte key to every frame. MIC with TKIP countermeasures eliminates the flaws in WEP at the cost of increased overhead and computation.

2 Analysis of Existing Protocols

In addition to the WEP weaknesses identified in section 1.2.2, there are a number of other issues with WEP and WPA 1.0 and 2.0 that are discussed below.

2.1 Questionable use of RC4

It is well known that WEP has been broken, and that tools exist on the Internet to decipher messages sent using WEP after reading as few as 20,000 messages. This decryption relies upon an improper use of RC4 [Mantin 2001]. Mantin identifies three significant RC4 weaknesses which, when RC4 is used improperly, make it trivial to attack. The most significant weakness is at the start of a cipher stream, which would let an attacker determine a key in 10^6 packets. WEP aggravates this problem by using a weak initialization vector and directly concatenating the IV to the shared key. WPA 1.0 still suffers from this RC4 weakness but uses extended IV, without direct concatenation to the key, to make it harder for an attacker to figure out the actual key.

This is not to say that RC4 is by nature susceptible to easy attacks. Beyond the first few bytes, RC4 encryption streams are excellent pseudo-random streams which are well suited to cryptography.

2.2 Unencrypted Control and Management Messages

All of the public encryption schemes for IEEE802.11 use in-the-clear control, and management messages, such as acknowledgements, association request messages, authentication and deauthentication messages. This use can compromise the integrity of encrypted data, or can open the network to other forms of attack, such as replay attacks, modified message replay attacks and fake management messages when WEP is used. WPA 1.0 uses encrypted message integrity fields (MIC key) to identify fake messages, but at a cost of increased overhead and reduced performance.

2.3 Insider Attack

Insiders are a significant threat to security systems. Shared key systems like WEP and WPA-PSK give insiders open access to all messages exchanged and permit active attacks. More thought is needed to restrict the access of insiders.

2.4 Power/processing Costs or Time Costs

For IEEE802.11 nodes powered by a limited power supply (eg. batteries), more complex encryption equates to increased energy consumption and processing times. RC4-KSA carried out on every packet adds an extra processing cycle in WEP and WPA 1.0, while WPA's extra fields (TSC and MIC Key) and AES encryptions add even more processing.

Without these additional overheads however, RC4 is a very efficient strong algorithm, consisting of a single byte-swap and XOR for each target byte. It should be possible to design an efficient algorithm, which uses the strong properties of RC4 with minimal processing. SBKH is such an algorithm.

3 SBKH Protocol Overview

SBKH is a state-based encryption protocol in which two communicating nodes also share a common knowledge of the RC4 state. As discussed above, WEP and WPA reinitialize RC4 state for every packet and generate cipher stream from the initialized RC4 state. This is due to the change in the IV for every packet, which results in the change of the RC4 seed.

SBKH does not reinitialize RC4 states, rather it maintains the same RC4 seed for a duration known to a pair of communicating nodes. This will require the initialization of the RC4 state (running RC4-KSA) to be done only when the base key changes. After this, communicating nodes keep using the same cipher stream, following the stream together, byte-by-byte, to encrypt and decrypt packets exchanged between them.

SBKH also includes an offset that will be shared between a pair of communicating nodes. This offset represents the number of octets to run down the cipher stream after every initialization of the RC4 state by the pair of communicating nodes, whenever a key is changed. Only after running down the stream, can a node start to encrypt or decrypt any packets successfully.

This protocol has been designed to operate with existing hardware as much as possible with minimal changes to the firmware. This can be justified by the fact that there are already millions of 802.11 cards shipped and that a change in the hardware will not solve the security issues with these existing 802.11 cards.

4 SBKH Protocol Details

4.1 Terminology

SBKH introduces some new concepts not needed in other shared key cryptography.

Communicating Nodes: In a managed or basic service set (BSS) network, the communicating nodes indicate any node in that network and the access point (AP) of that network. In an ad hoc or independent basic service set (IBSS) network, the communicating nodes indicate any two nodes that belong to the same network and are communicating directly with each other.

State Synchronized: For a successful communication between two nodes A and B, the RC4 state corresponding to A and B must be the same for a message to be encrypted and decrypted successfully. This is referred to as State Synchronized. A pair of nodes will be State Synchronized by successfully encrypting and decrypting exactly same number of octets (since each message successfully decrypted

was also successfully encrypted). Hence, it is important to maintain two sets of RC4 states (one for each communication direction: uplink and downlink) for a pair of nodes to be State Synchronized.

SBKH requires sharing of parameters between communicating nodes. These shared parameters include Base Key Pair, Key Duration, RC4 states, Offset and a new explicit SBKH sequence counter (SSC).

Base Key Pair: A Base Key Pair consists of two full 128 bit keys (Base Key_U for uplink and Base Key_D for downlink), which will be used as RC4 seeds. These keys may be selected from a list called Base Key List, which may be common to all the users within the same BSS. The keys may also be per-session keys that are agreed between the two communicating nodes. The generation and distribution of the Base $Key_{U/D}$ and the Base Key List is out of scope of this paper, although in this paper, a strong key generation, distribution and selection algorithm is assumed.

Base Key List: A list of keys shared by all nodes in a network. Actual list composition is not relevant to this paper as long as the keys are strong. Techniques such as portioning 128-bit Base Key into a 96-bit primary and a 32-bit secondary key and changing the secondary key (based on Timestamp, incrementing, look-up table, etc.) for an implicit Base Key List could be used.

Key Duration: Key Duration indicates how often a base key pair has to be changed or when a base key pair changes. This protocol relies on the fact that the timestamp from the beacon gives a common notion of time within that network and hence can be used with Key Duration to have common knowledge of when to change a key between any pair of nodes. Change of Base Key Pair may be just as easy as selecting the next key pair from the Base Key List.

Offset: SBKH defines one set of offsets called Initial Offset ($I\text{-}Offset_U$ for uplink and for $I\text{-}Offset_D$ downlink) which are used to indicate how far down the cipher stream should a node start encrypting and decrypting messages for a given Base $Key_{U/D}$. This is referred to as running down the cipher stream. Running down the cipher stream for $I\text{-}Offset_{U/D}$ number of octets will happen only whenever a key rollover takes place. The purpose of this offset is to discard $I\text{-}Offset_{U/D}$ number of octets from the start of a stream (carried out only once for any Base Key), strengthening RC4.

SBKH Sequence Counter (SSC): SBKH uses the 24-bit IV field within original WEP data frames of 802.11 (refer Fig. 1) as a SBKH Sequence Counter (SSC) field that will be maintained separately for each direction (SSC_U, SSC_D) for a pair of communicating nodes. Hence, SSC is different from 802.11 MAC's sequence number, which is maintained by a transmitter for the whole network and not for a pair of nodes. The ability to maintain a pair-wise sequence counter will lead to an easy decision making while trying to decrypt an incoming SBKH-encrypted data packet.

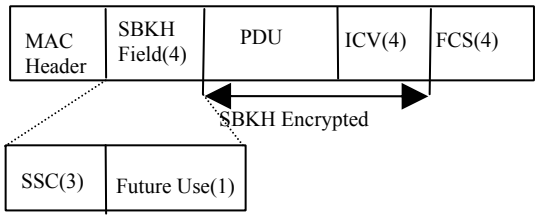

Fig. 1 SBKH Encrypted Frame

RC4 States: RC4 state is a state array with 256 state elements and two indices. Each state element and each index is of 8 bits in length, making the overall RC4 state to be of 258 octets in length. SBKH defines four pairs of RC4 states: Initial RC4 States ($IRC4_{U,}$ $IRC4_D$), Previous RC4 States ($PRC4_{U,}$ $PRC4_D$), Current RC4 States ($CRC4_{U,}$ $CRC4_D$) and Next RC4 States ($NRC4_{U,}$ $NRC4_D$). The notation may be extended, so that $CRC4_{U,j,B}$ corresponds to the state in the receiver for $CRC4_U$ for packet j as maintained by node B and SSC_U = j mod(2^{24}-1). A state designation without the subscript refers to both uplink and downlink variables.

IRC4 are (collectively) the RC4 states after performing RC4-KSA and RC4-PRGA for I-$Offset_U$ or I-$Offset_D$ number of octets for every Base Key Pair. A node may start encrypting data packets with a new Base Key only after calculating $IRC4_U$ and $IRC4_D$ for the corresponding Base Key_U and Base Key_D respectively. $IRC4_U$ and $IRC4_D$ are the RC4 states corresponding to the I-$Offset_U$ and I-$Offset_D$ for a given Base Key Pair.

PRC4 are the RC4 states corresponding to previously successfully transmitted or received and acknowledged SBKH encrypted packet. $PRC4_U$ and $PRC4_D$ are updated independently.

CRC4 are the RC4 states with which encryption and decryption of the subsequent packet takes place for a given Base Key. $CRC4_U$ and $CRC4_D$ are updated independently.

NRC4 are the RC4 states corresponding to I-Offset and I-$Offset_D$ for the next Base Key Pair. $NRC4_U$ and $NRC4_D$ are updated independently.

PRC4 and CRC4 states are continuously maintained for a pair of nodes: IRC4 and NRC4 states are logical states and may be generated as needed or maintained continuously, an option of interest to resource-limited devices. If a node is memory limited but not power or time limited, CRC4 may also be generated from PRC4 at a small performance penalty.

4.2 Basic Protocol Operation

Communication begins after initialization of SSC_U and SSC_D to zero, $PRC4_U$ = $CRC4_U$ = $IRC4_U$ and $PRC4_D$ = $CRC4_D$ = $IRC4_D$.

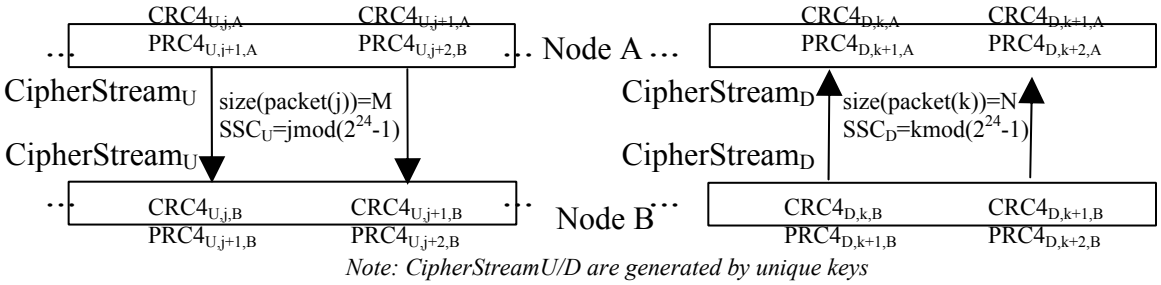

Fig. 2 SBKH State Synchronization

In Fig. 2, two nodes A and B are exchanging packets encrypted based on a Base Key Pair shared between A and B. A sends packet j in its uplink encrypted at $CRC4_{U,j,A}$ to B. After receiving packet j, B compares $SSC_{U,j}$ with its $SSC_{U,j-1}$ which according to B was the last successfully acknowledged

packet's SSC_U. If $SSC_{U,j}$ is one more than $SSC_{U,j-1}$, then B decrypts packet j at $CRC4_{U,j,B}$. After successful decryption ($CRC4_{U,j,B} = CRC4_{U,j,A}$) B acknowledges the packet to A and also updates its SSC_U to $SSC_{U,j}$. B then updates its $PRC4_{U,j+1,B} = CRC4_{U,j,B}$ and its $CRC4_U = CRC4_{U,j+1,B}$, the state where decryption of the packet corresponding to $SSC_{U,j+1}$ will begin. After the receipt of B's acknowledgment, A updates its $PRC4_{U,j+1,A} = CRC4_{U,j,A}$ and its $CRC4_U = CRC4_{U,j+1,A}$, the state where encryption of the packet with $SSC_U = j+1$ will begin . A also updates $SSC_U = SSC_{U,j+1}$ which will be used in the subsequent packet with $SSC_U = j+1$. The same discussion applies for packets sent by B to A on its downlink.

4.2.1 Retransmissions and Packet Drops

If a packet or fragment times out and is dropped, the subsequent packet or fragment has the same SSC as the dropped packet or fragment and the transmitter does not update PRC4 and CRC4. Hence, encryption of the subsequent packet begins from the same state as that of the dropped packet.

If the transmission isn't a retry and if the previously acknowledged SSC ($SSC_{U,j}$) and the received SSC from A ($SSC_{U,k}$) differ by more than 1 (i.e. k>j+1), then B may drop the packet without acknowledgment or may initiate resynchronization.

If the transmission of packet j is a retry (retry field in MAC frame set to 1), and if B previously acknowledged packet j and updated $CRC4_U$ to $CRC4_{U,j+1,B}$, and $PRC4_U$ to $PRC4_{U,j+1,B}$ (= $CRC4_{U,j,B}$), then B decrypts the packet from $PRC4_{U,j+1,B}$ or could optimize by sending an acknowledgment without re-decrypting.

If the transmission of packet j is a retry and if B previously did not receive or acknowledge this packet, then B decrypts that packet using $CRC4_{U,j,B}$ as it would do for any new packet with expected SSC. If the decryption is successful, node B will send acknowledgement to node A.

If the transmission is not a retry and if the previously acknowledged SSC (SSC_U) and the transmitted SSC ($SSC_{U,j}$) are the same, then B identifies that acknowledgment of the previously transmitted packet was not received by A and the packet was dropped after retries. B decrypts the new packet using RC4 state $PRC4_{U,j+1,B}$ since A encrypted the packet at $CRC4_{U,j,A} = PRC4_{U,j+1,B.}$ After decrypting the packet and acknowledging it, B updates $CRC4_U$ to $CRC4_{U,j+1,B}$, the state immediately following the last byte decrypted and leaves $PRC4_{U,j+1,B}$ unchanged.

4.2.2 Key Hopping

Two nodes communicating with each other remain State Synchronized as mentioned in section [4.2] if the Base Key Pair has not changed. If the Key Duration parameter indicates time to change the Base Key Pair, the transmitter starts encrypting packets and fragments using the new Base $Key_{U/D}$ following the key change.

For the following discussion refer to Fig. 2 and assume that A identified a need for key change before encrypting packet j. The discussion only considers A and B updating Base $Key_{U.}$ The same discussion applies to update Base Key_D for B sending packets to A on downlink with B and A interchanged.

A calculates $IRC4_U$ based on the new Base Key_U, and updates $PRC4_{U,j,A}$ and $CRC4_{U,j,A}$ to $IRC4_U$ after the successful acknowledgment of packet j-1. A then continues encryption of packet j based on the new Base Key_U. When B identifies time to change Base Key_U, it calculates $NRC4_D$ and $NRC4_U$, but does not immediately update $PRC4_{U,j,B}$ and $CRC4_{U,j,B}$ based on the new Base Key_U. B keeps

decrypting subsequent packets based on the old Base Key_U until the decryption fails once, and then tries decryption with $NRC4_U$ (note that for some circumstances B can optimize and try $NRC4_U$ first or decrypt both in parallel). The decryption succeeds, and B updates its $PRC4_U$ as $PRC4_{U,j+1,B} = NRC4_U$ and its $CRC4_{U,j+1,B}$ as the state where the decryption of the subsequent packet (j+1) will begin based on the new Base Key_U. B then clears $NRC4_U$. Following this, encryption and decryption of subsequent packets exchanged follow the discussion in section [4.2] until the next key change.

4.3 Support for Broadcasting and Multicasting

The lack of acknowledgment for broadcasting and multicasting packets in 802.11 makes maintenance of synchronization harder, but not impossible. Future research in SBKH will investigate state exchange using other resynchronization techniques, should state be lost. For now, broadcast and multicast packets can use the less secure WPA with a key different from the Base $Key_{U/D}$ being used for data packets to avoid any possible discovery of the SBKH Base Keys.

4.4 Working with EAP-based Networks

This protocol can be easily integrated with EAP-based networks although it targets low power sensor networks and SOHO users. This will allow nodes within a network to negotiate per-session keys. The negotiated per-session key will be the Base $Key_{U/D}$ for that session. The nodes may also negotiate Key Duration and I-Offset for every session. The details of interoperability with existing EAP protocols will be discussed in the future papers.

4.5 Support for DCF and PCF based WLANs

The 802.11 standard specifies two types of coordination functions: distributed coordination function (DCF) and point coordination function (PCF). Under DCF, medium access is carried out by medium access with collision avoidance (MACA) protocol, and sometimes with request to send (RTS) and clear to send (CTS). Under PCF, medium access is carried out using polling. This proposal will work with both DCF and PCF based 802.11 networks.

4.6 Support for Power-Save Nodes

The 802.11 standard supports nodes to be under power-save mode. Under this mode, a node wakes up to listen to specific beacons to check for any buffered packets for itself and receives the buffered packets, if any.

SBKH imposes a requirement to support such power save nodes. The requirement is that the Key Duration of a Base Key be at least twice as the wake-up duration of any power save node in that network. This will make sure that power save nodes wake up at least twice before the Base Key changes. This requirement is imposed to avoid buffered packets being encrypted with out-dated Base Keys, which will result in unsuccessful decryption of such packets.

4.7 Support for IBSS

The 802.11 standard also supports ad hoc networking in which nodes are allowed to send packets to each other directly. This type of network is called an IBSS network. In this mode, the beacon generation is distributed and so all participating nodes are allowed to send beacons, one at a time. Although beacon generation is distributed, all nodes within an IBSS share a common notion of time.

The other major implication that IBSS has for SBKH is that each node communicates with a number of neighboring nodes, and must maintain RC4 states for each node with which it forms a communicating pair. This means that selection of keys from a Base Key List and Key Duration also must be negotiated and maintained for each communicating pair as well.

Some IBSS networks have a dramatic simplification which can ease this process. If the data being exchanged does not have readily identifiable portions, such as TCP/IP headers or fields of fixed patterns, then an IBSS could use a single Base Key Pair, the same Base Key List (if any), the same I-Offset, and the same Key Duration. We note that the RC4 state will differ between different communicating pairs very soon after key hop because of the different sizes of packets and communication rates, so only insiders which know the Base Key Pair, Key Duration, and number of octets exchanged between communicating nodes of interest can decrypt packets. We also note that this simplification does not eliminate the need to maintain states PRC4 and CRC4 for all nodes which are communicating nodes with us.

Other schemes for key generation are needed for an IBSS with patterned data, since reused cipher streams permit data recovery when some of the data is known. Alternative schemes include:

a) indexing into the base key list using the communicating pair's MAC addresses and some parameter based upon beacon timestamp to select a unique pair of keys,
b) maintaining a single shared key portion of appropriate length, say 96 bits, and generate a unique portion (of 32 bits) based upon a changing criteria, such as a relevant portion of the beacon timestamp or a number which increments with the beacon at some frequency small enough to guarantee that each association created by communicating pairs generates a unique key,
c) negotiating Base Keys or session-oriented Base Keys and other SBKH parameters, using 802.1x, EAP, or other techniques.

Once the first Base Key has been allocated, key hop may occur by the normal process of selecting the "next" key from the Base Key List (if a key list exists), or by repeating the key generating process with new key fragments.

Broadcasting and multicasting within an IBSS can be carried out using techniques proposed under section [4.3].

4.8 Support for Handoff

Handoff between two access points and two IBSSs is also supported under the proposed scheme. A simplistic approach to handle handoff would be to have same Base Key Pair, same Base Key List (if any), and same Offset between the networks (with no identifiable portions in the data such as TCP/IP headers) of interest. This way the handoff between such networks may be quick and easy. Other techniques such as the ones listed in section 4.7 may also be considered for networks with or without identifiable portions within data.

Of course different networks having different SBKH shared parameters may need negotiations based on inter access point protocol (IAPP) and/or 802.1x [Standard 2001] and EAP based protocols. The details of such implementations are out-of-scope of this paper and may be included in the future articles.

5 Analysis of Protocol

Now that we have presented our protocol, it is important to compare it to the criteria that we have set forth in sections [1 and 2].

5.1 Use of RC4

SBKH uses RC4 in a way which makes effective use of RC4's strengths and avoids most of its weaknesses. Instead of the problematic stateless approach, a single RC4 encryption stream is followed for multiple packets for each communication direction of each pair of communicating nodes, starting at a nonzero offset known only to the communicating pair: and without exchanging key-specific or state-specific knowledge.

Because SBKH shares state, communicating nodes must maintain state and follow specific protocols to ensure the states remain synchronized. By using RC4 in this way, SBKH avoids known initialization weaknesses of RC4.

Pair wise independence of communication encryption creates a strong encryption protocol, even if the listener is an insider, while maintaining synchronization between communicating pairs. We have shown through model checking of the protocol that senders and receivers can stay synchronized, except for situations involving hard shutdown where state may become lost or an active interloper forcing loss of synchronizations. For such situations we have a resynchronization protocol that considers any desynchronizations that may occur.

5.2 Unencrypted Control and Management Packets

Since WEP and WPA do not encrypt the packet headers, or control and management packets, SBKH cannot encrypt such packets and maintain backward compatibility. SBKH will instead add encrypted payload portions to authentication and association based management messages (not discussed in this paper), and thus will eliminate denial of service attacks based on spoofing of such management messages. SBKH ignores man-in-the-middle attacks because these wireless systems assume that each communicating node can always hear the other node.

5.2.1 Denial of Service Attacks

SBKH is much less susceptible to denial service attacks than are either WEP or WPA-PSK, since more of the security protocol is private to the parties, as follows:

(i) **Fake (dis)association**: SBKH will have association based messages properly encrypted using the same strong scheme as data messages; hence any such in-the-clear messages will be ignored or recorded as an active attack.

(ii) **Fake (de)authentication**: SBKH will also have authentication based messages contain encrypted portions to avoid spoofing of such messages.

5.3 Replay attacks

Replay attacks of encrypted packets assume that the decryption stream is still valid. Under SBKH, the decryption stream has moved and any replayed packets will fail encryption validity checks (ICV) and will be noted as an active attack, making replay attacks useless.

5.4 Modified Packet Attacks (Forgery and Bit-Flipping attacks)

Modified packet attacks take two forms, header modifications and encrypted-body bit flipping. Both forms of attack will fail for the same reasons as the replay attacks, discussed above.

Of course, other forms of denial of service, such as swamping the channel with white noise or useless packets will always interfere with wireless communication, but these attacks will not aid an interloper to decipher packets or to join a BSS.

5.5 Power/processing Costs or Time Costs

SBKH is less expensive in terms of power and CPU resources of the transmitter/receiver than is WEP, and is significantly cheaper than WPA. We modeled the WEP-based RC4 encryptions and SBKH encryption scheme on standard desktop and workstation computers, predicting a 50% reduction over WEP, a 75% reduction over WPA and a 70% reduction over AES, in encryption processing for 200-byte packets.

5.6 No Key-change Knowledge in SBKH

With SBKH, almost all knowledge of key indexes (eg. IV in WEP or WPA), initializations and authorizations, and key changes are implicit in the protocol and cannot be determined by an analysis of in-the-clear messages. Specifically,

(i) The first few octets are discarded and encryption begins at $IRC4_{U/D}$ in the encryption stream;
(ii) Key-hopping depends only upon a private Key Duration parameter known only to the communicating pair.

By using this approach, it is computationally hard for an interloper to find the decryption location, even if they had the same Base Key.

5.7 Verification of SBKH

Since SBKH is a pair-wise state-based protocol, each node must maintain an exact copy of the RC4 state so that the next set of encryption octets can be matched. If a node gets out of synchronization by even a single octet, then encryption synchronization is lost and recovery requires a resynchronization protocol.

The challenge, therefore, is to verify that SBKH nodes keep the same place in the encryption chain even when corrupted packets, timeouts, retransmits, and key rolling occur.

To verify the correct operation of SBKH, we subjected significant portions of the protocol to formal verification using the Promela formal specification language and the SPIN model checker [SPIN 2003]. This approach performs static analysis of all of the interleavings of the two nodes encrypting messages via a state-based encryption and exchanging those messages via a medium. We successfully modeled message corruption and retransmission as well as behavior at key rolling and confirmed that the protocol is robust over these domains. Through this analysis we confirmed that two encryption states must be maintained by each node in each direction and that a pairwise message counter (SSC) improves efficiency and eliminates retransmissions due to wrong state selection.

We also determined through this process that an active attacker could send false acknowledgements for corrupted packets to one node and force it to lose encryption synchronization. We therefore needed

to develop a resynchronization protocol, although we suspect that implementation errors and catastrophic node shutdowns could also result in loss of encryption synchronization and require a resynchronization protocol. The resynchronization protocol will be the subject of another paper.

5.8 Implementation Complexity

The protocol proposed is both simpler and more complex than existing protocols, such as WEP or WPA. The protocol is simpler in that RC4-KSA is not required on the creation or reception of every packet. This translates directly to some simplification and likely power saving (for battery-operated systems) for systems that use SBKH.

The protocol is more complex in that more state is required to be maintained for decryption, specifically each non-access point node must maintain 4 encryption RC4 states for each node, which is communicating with it. Each encryption state consists of 258 octets plus possible ancillary information. This additional storage should amount to a very few extra kilobytes for every directly communicating node. We do acknowledge the need for encrypted management message such as a Disassociation message to indicate release of resources tied to a node that is leaving the network.

We have also considered likely hardware support for RC4, and believe that it should be easy to selectively generate RC4 state from a Base $Key_{U/D}$ or use a pre-generated RC4 state. Hence, we believe that this protocol should be compatible with existing hardware.

All things considered, we believe that this protocol is very competitive with existing encryption technologies.

5.9 Implementation Experience

We have implemented the encryption scheme using standard RC4 libraries on worstation-class processors, and have modeled the encryption, transmission, reception, decryption, and key-hopping parts of the protocol using a model checker. Some issues that we uncovered led us to propose a formal association/disassociation and a resynchronization phase. The issues uncovered were not with the basic protocol, but with a consideration of implementation errors or some possible active attacks. The basic protocol as described herein, we believe is robust, internally consistent, and efficient.

6 Conclusions

SBKH implements a novel encryption for wireless systems and provides strong encryption security without additional overheads of encryption and processing cost, and performance reduction. Hence, SBKH is well suited for battery-operated 802.11 devices and SOHO users. SBKH has full support for different 802.11 modes such as DCF, PCF, IBSS and power save (PS). SBKH can also be extended to 802.1x and EAP based networks with ease.

7 Future Work

There are a few areas associated with this protocol, which need further investigation:
(i) The formal model using the model checker has served us well, but is near its limits in computability. An alternative approach is to use a theorem proving approach to validate the protocol. An implementation on real hardware is also required.
(ii) Association message formats, ad-hoc mode and broadcasting require validation and possible further exploration.

(iii) A strong key generation and distribution recommendation would be very useful.
(iv) Encryption of parts of payload within Authentication and Association messages.
(v) SBKH support for broadcasting and multicasting frames.
(vi) Optimization of required memory space to successfully run SBKH is also a topic for future research.

8 Bibliography

[Draft 2003] Draft Amendment to Standard For Telecommunications and Information Exchange Between Systems – LAN/MAN Specific Requirements – Part 11: Wireless Medium Access Control (MAC) and physical layer (PHY) specifications: MAC Security Enhancements, IEEE Std. 802.11i/D7.0, Oct. 2003.

[FM 2000] Fluhrer S. and McGrew D., Statistical Analysis of the Alleged RC4 Keystream Generator, FSE: Fast Software Encryption, FSE2000, Springer-Verlag, 2000.

[FMS 2001] Fluhrer S., Mantin I., Shamir I., Weaknesses in the key scheduling algorithm of RC4, SAC'2001, 2001.

[Mantin 2001] Mantin I., Analysis of the Stream Cipher RC4. Weizmann Institute of Science, Nov. 2001.

[Moskowitz 2003] Moskowitz R., Simple Secrets/Simple Security, ICSA Labs, 2003.

[Roos 1995] Roos A., A Class of Weak Keys in the RC4 Stream Cipher. sci.crypt posting, Sept. 1995.

[Standard 2001] IEEE Standard for Local and metropolitan area networks – Port-Based Network Access Control, Oct. 2001.

[SIR 2001] Stubblefield A., Ioannidis J., and Rubin A. D., Using the Fluhrer, Mantin, and Shamir Attack to Break WEP, AT&T Labs Technical Report TD-4ZCPZZ, Aug. 2001.

[SPIN 2003] The SPIN Model Checker: Primer and Reference Manual, Addison Wesley, 2003.

[Walker 2002] Walker J., 802.11 Security Series – Part II: The Temporal Key Integrity Protocol (TKIP), Intel Corporation, 2002.