The 5th LCN Workshop on Security in Communications Networks (SICK 2009)
Zürich, Switzerland; 20-23 October 2009

A Frame Handler Module for A Side-Channel in Mobile Ad Hoc Networks

Marvin Odor, Babak Nasri
University of Ontario Institute of Technology
Oshawa, Ontario, Canada
{marvin.odor,bnasri}@gmail.com

Mazda Salmanian, Peter C. Mason
Defence R&D Canada (DRDC)
Ottawa, Ontario, Canada
{mazda.salmanian,peter.mason}@drdc-rddc.gc.ca

Miguel Vargas Martin, Ramiro Liscano
University of Ontario Institute of Technology
Oshawa, Ontario, Canada
{miguel.vargasmartin,ramiro.liscano}@uoit

Abstract— **In this paper, we establish a hidden 802.11 wireless channel, with the masking of the channel achieved by inserting intentional errors in the Frame Check Sequence (FCS). We design a frame handler module to provide a proof-of-concept model of the side-channel using MATLAB and Simulink with Communication Toolbox. We justify using MATLAB over the other simulation tools because of its existing functions: physical layer IEEE 802.11 wireless local area networking (WLAN) standard, existing modular channel fading models, the MAC layer cyclic redundancy checksum (CRC) generator, the CRC Syndrome detector, and the capability of modifying fields in a frame. These existing functions allow for the creation of a frame handler which generates frames, according to our design, to be inserted as erroneous frames and recovers frames from normal 802.11 traffic. Herein we provide the design and details of the implementation of the channel. Our design offers the ability to introduce error detection and correction capabilities, and protection against passive monitoring defences. This simulation framework is a step towards the development of more sophisticated environments including multi-node simulations that maintain robust and reliable side-channel communication.**

Keywords— Side channels, Mobile ad hoc networks (MANET), Medium Access Control (MAC), Cyclic Redundancy Checksum (CRC), network security.

I. INTRODUCTION

WIRELESS networks are designed to tolerate errors, measured by bit or frame error rates: BER (10^{-3} – 10^{-7}) and FER (1% - 3%). Errors are generally caused by fluctuations of the signal strength through the medium, known as fading and shadowing. These types of variations are inherent in a Mobile Ad hoc Network (MANET) environment, meaning that MANETs can be expected to have measurable FERs even when operating in seemingly ideal conditions. The Physical (PHY) and Medium Access Control (MAC) layers are designed, in almost all wireless protocols, to detect corrupted frames. Using a cyclic redundancy check (CRC) on the frame payload and appending information derived from this CRC into a Frame Check Sequence (FCS) field is a common technique for checking its integrity at the receiver. The CRC functions like a hash value of the payload contents so the receiver should be able to take the payload and verify that performing the same hashing operation yields the same value in the FCS field.[1] If the values do not match, the frame is dropped. The handling of error differs among different wireless protocols. For example, military grade wireless protocols may be capable of a correcting certain level of error, whereas commercial wireless protocols, such as the IEEE 802.11 Wireless Local Area Networks (WLAN), discard the frame and ask / wait for retransmissions.

In this paper, we detail how to establish a side-channel by intentionally corrupting the value in the FCS field, shown in Fig. 1, of frames of our choosing. These frames will appear in error to stations not privy to the implementation of the side-channel, and will thus possess a certain degree of obscurity[2] as they lie hidden amongst the naturally occurring error frames typical of a MANET environment. This idea was first suggested for use in WLANs by K. Szczypiroski under Hidden Communications System for Corrupted Networks (HICCUPS) [1], but it was not implemented for lack of access to the MAC layer code of the modem. A form of a Denial of Service (DoS) attack called FCS False Blocking [2] also depends on modifications to the FCS value and the mitigation of this attack demonstrates why MANETs, as opposed to WLANs, are a more suitable choice for the implementation of such a channel [2].

In our implementation, we show how a judicious choice of the method for creating the side-channel frames can both reduce the probability of the channel being discovered by passive observers [3] and improve the throughput of the channel by providing error-correction capabilities. We demonstrate this capability using MATLAB and Simulink ™ and discuss why this simulation environment was chosen for the project at this stage of development. Preliminary work done by Defence R&D Canada and the Communications

[1] In general, a CRC is not strictly a hash function but many can be implemented as such and the mapping principles are analogous.
[2] We also refer to the side-channel as a hidden channel.

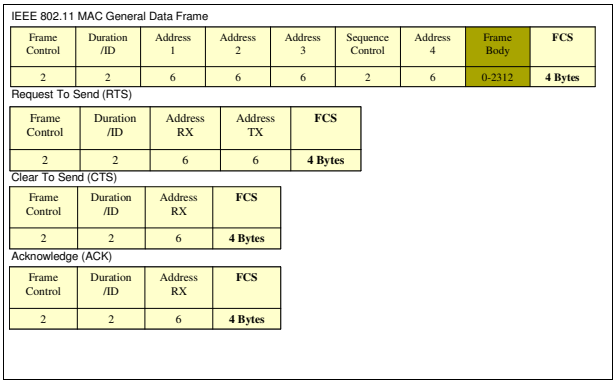

Fig. 1. Frame Check Sequence (FCS) is a hash value of the frame contents, regardless of the frame. The frames shown are used in the IEEE 802.11 wireless local area networking protocol.

Research Centre, Canada has shown that throughput on the order of 10 Kbps can be reasonably expected by such a side-channel when the FER of a MANET is changed from 1% to 1.5% by the insertion of the side-channel traffic. This amount of throughput could have high potential for MANETs, which adds an extra layer of ambiguity to a traffic flow even when encrypted.

The rest of this paper is organized as follows. Following this introduction, we present the methodology and design of our side-channel in Section2. In Section 3, we provide the implementation of our frame handler in MATLAB Simulink. We conclude with a discussion in Section 4. Pseudo code for the implementation is provided in the Appendix.

II. Methodology

A hash function can provide a short fingerprint [4] of its (usually longer) input data, which can be used to check the data's integrity. If the data is changed or altered, its fingerprint also changes. Like most existing wireless protocols, IEEE 802.11 uses a Cyclic Redundancy Checksum (CRC) function for integrity checking in the Media Access Control (MAC) layer. The IEEE 802.11 standard [5] dictates using the CCITT CRC-32 polynomial for the MAC header and frame body of General frames (data frames), and Control frames [Request To Send (RTS), Clear To Send (CTS), and Acknowledge (ACK)]. As shown in Fig. 1, these frame types include a 4-byte-long Frame Check Sequence (FCS) field at the end of their frames which is filled with the remainder of a long division operation – dividing the data by the CRC-32 generator function – which is then the fingerprint of the MAC header and frame body. At the receiver, this operation is performed again and the resulting remainder is compared to that of the FCS field in the received frame. If the calculated remainder of the long division matches the contents of the FCS field, then the MAC layer approves the frame and sends it to upper layers for further processing. Otherwise, the frame is found to be erroneous and is discarded. The MAC layer may rely on the Automatic Repeat Request (ARQ) to replace the discarded frame and to deliver it in sequence to upper layers. Upper layers, such as TCP, depend on their own timeouts for retransmissions and packet delivery in sequence to the Application layer.

As discussed in the introduction, we wish to establish a side-channel by altering the FCS field intentionally at the transmitter so the frames can be made to appear with error for all receivers except for those who know to how to process the erroneous frames further by filtering them rather than dropping them. Let us denote the subset of nodes in the network that are "aware" of the side-channel as *SideChannel* nodes and all other nodes as *Normal* nodes. When a frame is received by any node, its integrity is checked at the MAC layer according to the condition:

$$\Delta FCS = CRC_{calculated} \oplus FCS_{Rx} \quad (1)$$

Where $CRC_{calculated}$ is the remainder calculated over the received frame payload and FCS_{Rx} is the received value similarly calculated by the transmitter before sending the frame. Based on Eq. (1), the following four conditions are sufficient for establishing the side-channel:

i) $= 0$ ➔ frame is processed by *all* nodes
ii) $\neq 0$ ➔ frame is dropped by *Normal* nodes

For case ii) the frame is considered in error by *Normal* nodes, but the *SideChannel* nodes will now invoke their own filter on these frames:

iii) $= \phi(x)$ ➔ frame is processed by *SideChannel* nodes
iv) $\neq \phi(x)$ ➔ frame is dropped by *SideChannel* nodes

Where the function ϕ calculated over the payload x has been privately agreed upon by the *SideChannel* nodes. The choice of ϕ will be discussed in Section III.

To assist with our discussion, the partitioning of the wireless channel is demonstrated pictorially in Fig. 2, from which we extract and define some terminology. The wireless channel consists of the *Usable* channel and the *Error* channel (depicted with striped green and red lines respectively in Fig. 2). The *Error* channel is a naturally occurring manifestation of the coding techniques used for transmitting the information through the noisy wireless channel. All frames in the *Error* channel are dropped by *Normal* nodes, a property that provides our side-channel (shown in blue in Fig. 2) a measure of obscurity. Since the *Error* channel cannot be manipulated - it is a property of the wireless channel - our side-channel must occupy bandwidth in the *Usable* channel and will also be subject to transmission errors just like any other frame sent through the wireless channel. That is, the side-channel will spill into the *Error* channel as shown in the figure. If the same coding is used, the ratio of *Error*-to-*Usable* bandwidth in the side-channel will reflect that of the entire *Error*-to-*Usable* bandwidth in the wireless channel.

Since the communication bandwidth achieved by this side-

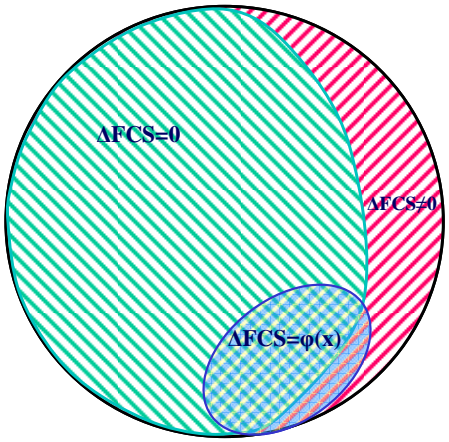

Fig. 2. A Venn diagram of our proposed side-channel with respect to the wireless channel as a whole. The green-striped area is the *Usable* channel, the red-striped area is the *Error* channel, and the blue-partitioned area is the side-channel.

channel uses bandwidth in the *Usable* channel, it cannot be rendered entirely covert. Because the side-channel is intentionally creating frames that appear corrupted to *Normal* nodes; these nodes will observe a decrease in *Usable* bandwidth with a concomitant increase in the size of the *Error* channel. This interplay must be considered when the *SideChannel* nodes choose a desired bandwidth for the side-channel; anything above a certain threshold is likely to trigger some advanced attack-detection techniques [2].

III. Implementation Details

When implementing the side-channel, there are a number of points to be considered, many of which are discussed in an overview of covert channels at different layers done by SANS[3] [6]. In this project, we choose the MAC layer as the logical setting to establish a side-channel because MAC layer changes, such as modifications of the FCS, can be performed by software upgrades to the driver of some 802.11 wireless cards [7].[4]

A. Design and Objectives

There are several objectives we wish to meet with this side-channel. As with any communication link, we want to maximize the bandwidth while minimizing errors. We have an additional condition that we want to also minimize the probability that an observer will detect the existence of the side-channel.

There are a number of ways that an intrusion or attack detection system could identify and flag the existence of the side-channel; among them:

1) The erroneous frames appear to hold uncorrupted (and possibly valuable) information.
2) The observed FER, increased by side-channel usage, is abnormally high.
3) The CRC values in the FCS field of erroneous frames are statistically biased. That is, $\phi(x)$ is not uniformly distributed.
4) An observer applies various common functions to payloads of error frames and discovers that a large percentage of the payloads are mapped onto the FCS by $\phi(x)$. That is, the observer guesses $\phi(x)$.

The first flag can be avoided by having the side-channel employ a form of encryption to the payload of its frames. Since the *SideChannel* nodes, by definition, share knowledge that *Normal* nodes do not, encrypting side-channel frames with mechanisms (such as WPA) using private keys adds little additional overhead to the design. The second, third, and fourth flags will be dealt with simultaneously by a careful selection of the function $\phi(x)$.

As discussed, this side-channel occupies a portion of the *Usable* bandwidth of the wireless channel by making it appear to be part of the *Error* channel. While this is an inescapable feature of our implementation, the ability to detect the channel can be reduced. The CRC-32 function used to fill the FCS field in 802.11 frames provides no error correction. Since we are modifying the FCS field for our side-channel, it is sensible to take the opportunity to employ a better mechanism. Existing research on the CRC-32 function suggests that an improved CRC generator polynomial could offer better error detection and correction measures [8, 9]. We choose one of these functions and call it an *Enhanced CRC*. As depicted in Fig. 3, adding improved error detection and correction to the side-channel increases the side-channel bandwidth by reducing the error rate within it. The figure shows this effect as a translation of the boundary between the *Usable* side-channel into the bandwidth of the *Error* side-channel. This would allow us to decrease the size of the side-channel while maintaining a comparable bandwidth that would have been obtained had we continued using the CRC-32 function for the FCS - the smaller the side-channel, the lower the risk of detection.

There is no guarantee, however, that letting $\Delta FCS = \phi(x)$, prevents the side-channel from being flagged by an observer who analyses the distribution of erroneous FCS values.[5] Ideally, the values we insert into the FCS field should be as random (evenly distributed) as possible to maintain the appearance that the values are not correlated with the payload. There are many ways to achieve this; we suggest the following for the side-channel:

[3] www.SANS.org

[4] The physical layer (PHY) frames also use a CRC polynomial for integrity checking, however modifications of the PHY must take place at the time of manufacturing the codec's firmware in the network interface card (NIC).

[5] To look at a trivial example, an *Enhanced CRC* function that generated only even values for the FCS field would skew the 50/50 balance between odd and even values that would be expected in the field.

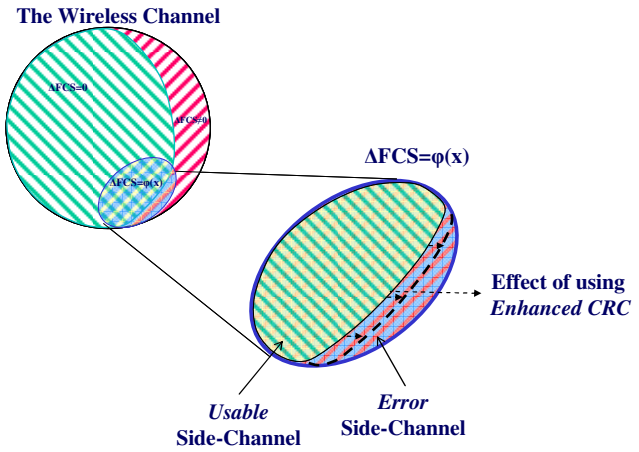

Fig. 3. A more detailed Venn diagram of our proposed side-channel with respect to its *Usable* side-channel and *Error* side-channel. Using an Enhanced CRC decreases the *Error*-to-*Usable* ratio of the side-channel compared to that of the wireless channel.

Let $\Delta FCS = \phi(x)$ such that:

$$\phi(x) = CRC_{enhanced} \oplus hash(Secret \mid Seq\# \mid MAC_{TxAddress}) \quad (2)$$

where *Secret* is a shared secret amongst the *SideChannel* nodes, *Seq#* is the sequence number of the frame, and $MAC_{TxAddress}$ is the MAC address of the transmitting node. The inclusion of the shared secret in the hash function input means that an observer who suspects that the *SideChannel* nodes are using $\phi(x) = CRC_{Enhanced}$ cannot distinguish between side-channel frames and *Error* channel frames because the FCS values have been masked by the hash output. The use of the sequence number guarantees that the argument of the hash function is unique for each frame sent by a particular *SideChannel* node. Since the output of a (good) hash function has near-uniform distribution, the FCS value used by the *SideChannel* nodes will be evenly distributed and, thus, uncorrelated to the payload from the perspective of an outside observer. The $MAC_{TxAddress}$ ensures that nodes using overlapping sequence numbers and sending some frames in common do not produce FCS collisions. Using Eq. (2), then, means that flags iii) and iv) listed above should no longer be a concern.

B. Simulation Considerations and Implementation

Our design objectives included the ability to create frames that could be appended with a CRC whose generator function could be modular. We required passing the frames through a wireless channel, also with modular statistical models such as Rayleigh, Rice or of our own creation. A private processor was required to receive the frames and filter those marked for private processing, those with modified CRCs that would otherwise be discarded. Parameters such as error rate and throughput must be measured throughout the simulation. A block diagram of such a design is provided in Fig. 4.

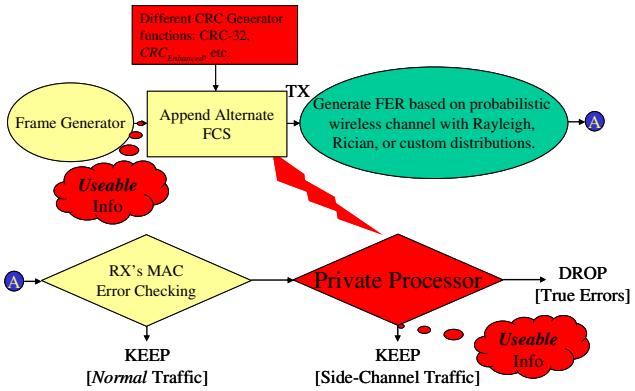

Fig. 4. A block diagram of our design requirements includes frame generation, calculation and appending FCS checksums, an error-producing wireless channel, and a receiver with a private processor for handling side-channel traffic that is aware of the CRC generation.

1) The simulator

For a proof-of-concept model, we considered several well-known simulation tools for developing and implementing the frame handler module for a MANET side-channel. We considered QualNet, NS-2, and MATLAB/Simulink because they are widely used by the research community for simulations that require wireless radio (physical) layer capability.

QualNet is a well supported simulation environment, equipped with an application programming interface (API), a programmer's guide that is continuously updated, and has an 802.11 physical and MAC layer simulation environment. The non-commercial version is not equipped with the Network Emulation Interface (IPNE). This library enables real data to be sent from one host to another, which we required in this study. In the source code of the physical layer, the function `Phy802_11CheckRxPacketError` uses a stochastic model to declare whether or not a frame is in error. It does not check the contents of the frame when making this decision since the simulator can deliver all packets without error. We need access to the frame itself to modify its contents and we would like to apply error-generation functions to each bit in the frame individually, as occurs in a true wireless channel. In addition, decisions on whether or not to drop a frame or process it further need to be made based upon its contents.

NS-2 has similar frame-error handling. Previous work at DRDC using NS-2's error function with a Rician channel model allowed us to demonstrate the Frame Error Rates (FERs) in a MANET scenario, but again it proves to be difficult to access the contents of the frame itself in order to manipulate the FCS, flip bits, and do error correction.

MATLAB has a Simulink tool equipped with IEEE 802.11b baseband physical layer standard [5] - all are part of the Communication Toolbox. The MATLAB Communication Toolbox and the Simulink package provide for a simulation

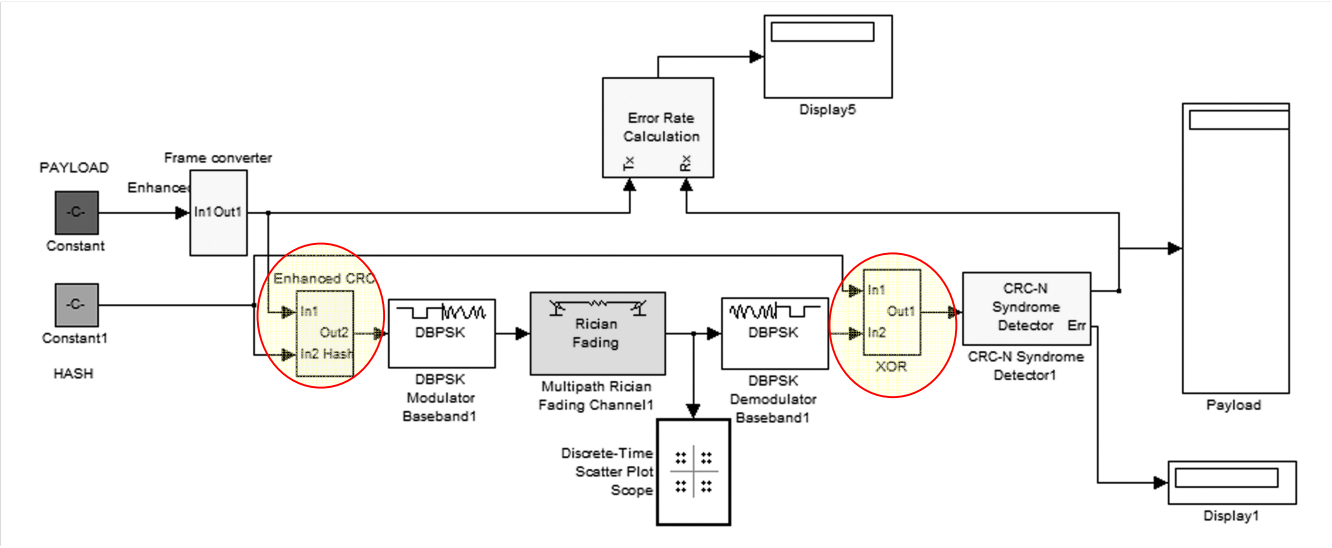

Fig. 5. A block diagram of MATLAB Simulink includes generation of payload in a frame with *Enhanced CRC* in its FCS field, sending the frame through a wireless channel with Rician model, frame reception, defragmentation and extraction of the frames with modified CRCs (XOR block) and private processing of the covert data (CRC-N Syndrome Detector). The side-channel is essentially established between the Enhanced CRC block and the XOR block.

environment where the transmitter, the wireless channel, and the receiver may be modeled separately in modular fashion, interconnected, and monitored. The Communication Toolbox includes CRC generators, error detection and correction techniques, multiple channel models, data communication sources, and modulation algorithms – all of which are needed for our proof-of-concept model. Other modules, such as Scope Display, Rate Detection, and BER Calculation may be added for monitoring and parameter measurements in the simulator. A careful test of the 802.11b Simulink model led us to conclude that this tool is currently the most suitable one for our simulations.

We used a set of Simulink blocks, including an IEEE 802.11 physical Layer, to implement our frame handler module [10], using the requirements mentioned above, shown in Fig. 5. Through the Simulink implementation we are able to generate payloads that are fragmented in frames with CRC checksums of our choosing. The frames are sent through a wireless channel modeled by Rician statistics. The XOR block performs signal reception and extracts the frames with modified CRCs while the CRC-N Syndrome Detector block performs private processing and defragmentation of the side-channel data. The highlighted Enhanced CRC and XOR blocks in Fig. 5 are explained further below. The pseudo code for the block diagram of Fig. 5 is provided in the appendix along with source codes for two functions: for reading a text file to be fragmented into frames and creating a hash in place of a CRC, as per Eq. 3.

2) Enhanced CRC and XOR blocks

As explained in Step 5 of the pseudo code in the Appendix, the Enhanced CRC block appends a deterministic value to the FCS field of the side channel frames. This deterministic value must be reproducible by the receiver at the XOR block to retrieve the modified frames, via Eqs. 1 and 2. As shown in Fig. 6, the Enhanced CRC block calculates the CRC for the frame and uses it to calculate an *Enhanced CRC* so that their difference satisfies Eq. 2. The *Enhanced CRC* gets appended to the frame.

The XOR block (Step 8 of pseudo code and Fig. 7 below) calculates the appended CRC of the received frame, as per normal MAC operation. However, if the calculated CRC does not equal the appended value in the frame and the frame is found in error, the XOR performs an extra verification as per Eq.2 to see if the frame were marked for private processing. If Eq. 2 is satisfied, the side channel frames are sent to the CRC-N Syndrome Detector block for further processing. Metric collection could also be implemented at this point if one wanted to test aspects of the additional error-correction provided by the enhanced CRC or see the effects of custom error-generating functions. We are investigating this with on-going work.

IV. Discussion and Conclusion

The side-channel in this proposal consists of frames intentionally made to appear in error. The intention of this content-hiding technique is to ensure that side-channel frames appear, to non-participating nodes or passive observers, as much as possible like errors caused by the wireless channel conditions. While this channel must, by design, alter the conditions of the wireless channel, much like the use of steganography alters the original image, steps may be taken to minimize detection.

One of the benefits of this proposed content-hiding capability is that it can be applied to any frame type in any

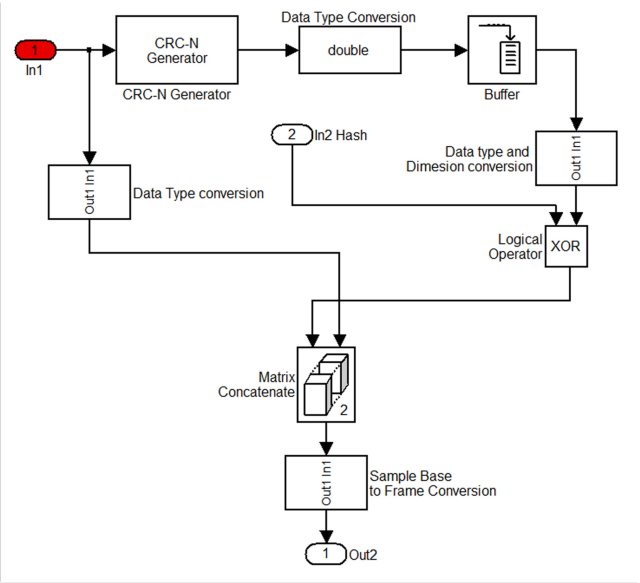

Fig. 6. A block diagram of MATLAB Simulink of Enhanced CRC block includes calculation of an *Enhanced CRC* based on Eq. 2 and 3 and appending it to the frame, instead of appending the frame's calculated CRC.

technology that uses some form of integrity checksum function. However, the applied technology must be chosen by considering the probability of detection of the hidden channel [11]. In wireless systems, such a hidden channel may be detected if the frame error rate (FER) increases abnormally beyond what is considered 'normal' under certain traffic conditions, terrain, obstructions and mobility. But the value of this application is that given those conditions in wireless channels, the range of what is considered 'normal' FER is wide – sometimes by as high as 25% [1, 6]. This property of wireless systems reduces the probability of detecting a hidden channel, because variations of FER – perhaps due to high traffic on the hidden channel – may be automatically attributed to (and hidden by) variations in wireless channel conditions rather than hidden channels.

A Mobile Ad hoc Network is an apt choice for this type of side-channel. Compared to wired networks, or even WLANS, the error channel in MANETs is large and highly fluctuating. Sources of these fluctuations include the mobility of both transmitter and receiver, the dynamic network topology and membership, and the diverse environmental conditions in which they may be deployed. As well, the current design of the channel is such that it only supports point-to-point communication, as there is no mechanism in place for forwarding frames without intermediary nodes reprocessing the frame.

We have demonstrated how a careful choice of the method of altering the FCS field in our side-channel frames can preserve the side-channel's throughput while reducing the probability of detection. This is done by using an improved CRC function that provided error-correction capabilities, allowing us to occupy a smaller proportion of the usable

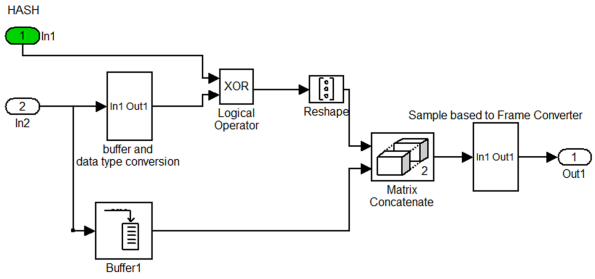

Fig. 7. A block diagram of MATLAB Simulink XOR block includes verification of the appended FCS with a calculated CRC as per Eq. 3. Once verified, then the frame is identified as covert and will be processed accordingly by the CRC-N Syndrome Detector.

channel for a given bandwidth. Mixing in additional parameters through a hash function protects against discovery by correlation-based statistical detection techniques.

On-going and future work in this project covers a number of areas. The MatLab/Simulink environment has proven to be an excellent starting point for designing the proof-of-concept frame handler for point-to-point communications and will continue to be a strong platform for testing different CRC functions and customized error-generating functions that may be applied to frames on a bit-by-bit level. As we move to a more dynamic networking environment to better simulate the conditions in a real MANET, we can begin the implementation of feedback mechanisms that may be implemented to control side-channel throughput, throttling it back when conditions are such that its use (more specifically, over-use) might trigger detection. We plan to also implement this side-channel capability on Smartphones and run our experiments in real-life conditions.

APPENDIX

A. *Pseudo Code*

1. Read message to be hidden (in our simulations we use a text file with a message Lorem Ipsum) into MATLAB and convert it into its binary representation.
2. Payload Constant: Use a callback function to import the binary data into Simulink (the Text File reader source code, see below)
3. In1Out1: Compute a CRC-32 in Simulink and appended to the payload.
4. Hash Constant1: In MATLAB, compute the MD2 hash of a secret message (this secret is securely pre-established and shared between the participants in the MANET equipped with a hidden communication channel) and convert it to its binary representation. Send this binary data to Simulink using a callback (the source code for hashing a text can be found in [12]).
5. Enhanced CRC: XOR the CRC-32 of Step 3 with the 32 least significant bits of the hash value of Step 4 (this operation calculates the ΔFCS as presented in Eq. 2). A

sub-blockset using this formula is implemented in Simulink and is shown in Fig. 6. The resulting ΔFCS replaces the CRC-32 of Step 3.

6. DBPSK Modulator Baseband1: To simulate the transmission of frames through the ether, we use a baseband modulator and a block with Rician fading channel.
7. DBPSK Demodulator Baseband1: The receiver side uses a baseband demodulator.
8. XOR: Before using Simulink's CRC-32 checker, we analyze the frame to see if it belongs to the hidden channel. We repeat Step 5 to verify whether the result of the XOR matches the hash value of the secret message, and if so then this frame belongs to the hidden channel. We append the CRC-32 to the frame for subsequent normal processing. We note that even hidden channel packets may be corrupted by the Rician fading block but this is what we would expect to happen in a real scenario. The XOR subsystem block is shown in Fig. 7.
9. Finally, frames are sent to a CRC-32 Syndrome detector block for private processing and defragmentation.

B. Source code for the Text File Reader

```
character_numbers = 4;

% Opens a file for reading
text_data = fopen('text.txt');

%Read text and assign it to t
t =
fscanf(text_data,'%c',character_numbers);

% Converts char decimal to binary
array2d = de2bi(uint8(t),8;

% check size of array to later reshape the
array
[m,n] = size(array2d);

%reshapes array from columns to rows
data_stream = reshape(array2d, 1, n* m);

% Send datastream to constant block in
simulink
set_param('inputcallback_mod/Constant','co
nstval',data_stream)
```

C. Source Code for hashing a text

```
h=hash('secret','MD2');
count = size(h);
hash_bin = de2bi(hex2dec(h(1:8)));
disp(hash_bin)
set_param('inputcallback_mod/Constant1','c
onstval',hash_bin)
```

REFERENCES

[1] Szczypiorski, K., HICCUPS: Hidden Communication System for Corrupted Networks, The Tenth International Multi-Conference on Advanced Computer Systems, Międzyzdroje, Poland, Oct 22-24, 2003, pp 31-40.

[2] Liang, S.T. and Weng, M.Y., Protecting IEEE 802.11 Wireless LANs against the FCS False Blocking Attack, Proceedings of the 11th International Conference on Parallel and Distributed Systems (ICPADS'05), 2005.

[3] Murdoch, S.J., Covert Channel Vulnerabilities in anonymity systems, University of Cambridge, December2007 http://www.cl.cam.ac.uk/techreports/UCAM-CL-TR-706.pdf

[4] Stinson, D. R., Cryptography: Theory and Practice, Chapman & Hall/CRC, 2006.

[5] IEEE Standard for information technology-Telecommunications and information exchange between systems-Local and metropolitan area networks-Specific requirements-Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications: Amendment 6: Medium Access Control (MAC) Security Enhancements, July 2004.

[6] Sbrusch, R., Network Covert Channels: Subversive Secrecy, As part of the Information Security Reading Room, © SANS Institute 2006.

[7] Neufeld, M., Fifield, J., Doerr, C., Sheth, A., and Grunwald, D., SoftMAC – Flexible Wireless Research Platform, Dept. of Computer Science University of Colorado, Boulder, November, 2005.

[8] Koopman, P. and Chakravarty, T., Cyclic Redundancy Code Polynomial Selection For Embedded Networks, International Conference on Dependable Systems and Networks (DSN), June 2004, http://www.ece.cmu.edu/~koopman/roses/dsn04/koopman04_crc_poly_embedded.pdf

[9] Koopman, P. 32-Bit Cyclic Redundancy Codes for Internet Applications, International Conference on Dependable Systems and Networks (DSN) http://www.ece.cmu.edu/~koopman/networks/dsn02/dsn02_koopman.pdf

[10] Odor, M., Vargas Martin, M., Liscano, R. , Salmanian, M. , Mason, P.C., A Confidential Wireless Channel for Side-Channel Communication in MANETs, Poster in the 30th IEEE Symposium on Security & Privacy, May 2009.

[11] Smith, R. W., On the Design of Network-Based Covert Communication Systems, Ph.D. Thesis Royal Military College of Canada, April, 2007, Supervised by Dr. Scott Knight.

[12] MATLAB Compute hash using MD2, MD5, SHA-1,SHA-256,SHA-384, and SHA-512 http://www.mathworks.com/matlabcentral/fileexchange/8944