Internet Engineering Task Force
Internet-Draft
Intended status: Informational
Expires: August 25, 2021

S. Yang
CUHK(SZ)
X. Huang
R.W. Yeung
CUHK
J.K. Zao
NCTU

February 21, 2021

BATS Coding Scheme for Multi-hop Data Transport
draft-irtf-nwcrg-bats-00

Abstract

BATS code is a class of efficient linear network coding scheme with a matrix generalization of fountain codes as the outer code, and batch-based linear network coding as the inner code. This document describes a baseline BATS coding scheme for communication through multi-hop networks, and discusses the related research issues towards a more sophisticated BATS coding scheme.

Status of This Memo

This Internet-Draft will expire on August 25, 2021.

Copyright Notice

Table of Contents

1. Introduction

This document specifies a baseline BATS code [Yang14] scheme for data delivery in multi-hop networks, and discusses the related research issues towards a more sophisticated scheme. The BATS code described here includes an outer code and an inner code. The outer code is a matrix generalization of fountain codes (see also the RapterQ code

described in RFC 6330 [RFC6330]), which inherits the advantages of reliability and efficiency and possesses the extra desirable property of being network coding compatible. The inner code, also called recoding, is formed by linear network coding for combating packet loss, improving the multicast efficiency, etc. A detailed design and analysis of BATS codes are provided in the BATS monograph [Yang17].

A BATS coding scheme can be applied in multi-hop networks formed by wireless communication links, which are inherently unreliable due to interference. Existing transport protocols like TCP use end-to-end retransmission, while network protocols such as IP might enable store-and-forward at the relays, so that packet loss would accumulate along the way.

A BATS coding scheme can be used for various data delivery applications like file transmission, video streaming over wireless multi-hop networks, etc. Different from traditional forward error correcting (FEC) schemes that are applied either hop-by-hop or end-to-end, the BATS coding scheme combines the end-to-end coding (the outer code) with certain hop-by-hop coding (the inner code), and hence can potentially achieve better performance.

The baseline coding scheme described here considers a network with multiple communication flows. For each flow, the source node encodes the data for transmission separately. Inside the network, however, it is possible to mix the packets from different flows for recoding. In this document, we describe a simple case where recoding is performed within each flow. Note that the same encoding/decoding scheme described here can be used with different recoding schemes as long as they follow the principle as we illustrate in this document.

The purpose of the baseline BATS coding scheme is twofold. First, it provides researchers and engineers a starting point for developing network communication applications/protocols based on BATS codes. Second, it helps to make the research issues more clear towards a sophisticated BATS code based network protocol. Important research directions include the security issues, congestion control and routing algorithms for BATS codes, etc.

1.1. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

2. Procedures

2.1. Introduction

A BATS coding scheme includes an outer code encoder (also called encoder), an inner code encoder (also called recoder) and a decoder. The BATS coding scheme can be used for a single data flow that includes a single source and one or multiple destinations. Thus there exists only one encoder with multiple recoders and decoders. The BATS coding scheme described in this document can be used by a Data Delivery Protocol (DDP) with the following procedures.

Outer Code Encoding at a source node which has the data for transmission:

* The DDP provides the data to be delivered and the related information to the BATS encoder.

* The BATS encoder generates a sequence of batches, each consisting of a set of coded packets and the information pertaining to the batch.

The batches generated at the source node are further recoded before transmitting:

* A BATS recoder generates recoded packets of a batch.

* The DDP forms and transmits the DDP packets using the batches and the corresponding batch information.

Recoding at an intermediate node that does not need the data:

* The DDP extracts the batches and the corresponding batch information from its received DDP packets.

* A BATS recoder generates recoded packets of a batch.

* The DDP forms and transmits DDP packets using the recoded packets and the corresponding batch information.

Decoding at a destination node that needs the data:

* The DDP extracts the batches and the corresponding batch information from its received DDP packets.

* A BATS decoder tries to recover the transmitted data using the received batches.

* The DDP sends the decoded data to the application that needs the data.

2.2. Data Delivery Procedures

Suppose that the DDP has F octets of data for transmission. We describe the procedures of one BATS session for transmitting the F octets. There is a limit on F of a single BATS session. If the total data has more than the limit, the data needs to be transmitted using multiple BATS sessions. The limit on F of a single BATS session depends on the MTU (maximum transmission unit) of the network, which MUST be known by the DDP. We have F is no more than (MTU-10)2^16-1 octets.

2.2.1. Source Node Data Partitioning and Padding

The DDP first determines the following parameters:

o Batch size (M): the number of coded packets in a batch.

o Recoding field size (q): the number of elements in the finite field for recoding. q is 2 or 2^8

o BATS payload size (TO): the number of payload octets in a BATS packet, including the coded data and the coefficient vector.

Based on the above parameters, the parameters T, O and K are calculated as follows:

o O: the number of octets of a coefficient vector, calculated as O = ceil(M*log2(q)/8).

o T: the number of data octets of a BATS packet, calculated as T = TO - O.

o K: number of source packets, calculated as K = floor(F/T)+1.

The data MUST be padded to have T*K octets, which will be partitioned into K source packets b[0], ..., b[K-1], each of T octets. In our padding scheme, b[0], ..., b[K-2] are filled with data bits, and b[K-1] is filled with the remaining data octets and padding octets. Let P = K*T-F denote the number of padding octets. We use b[K-1, 0], ..., b[K-1, T-P-1] to denote the T-P source octets and b[K-1, T-P], ..., b[K-1, T-1] to denote the P padding octets in b[K-1], respectively. The padding process is shown in Figure 1.

```
Z = T - P
Let bl be the last source packet b[K-1]
for i = 1, 2, ... do
  if Z + i >= T - 1 do
      bl[Z...T-1] = i
      break
  bl[Z...Z+i-1] = i
  Z = Z + i
```

Figure 1: Data Padding Process

2.2.2. Source Node Outer Code Encoding Procedure

The DDP provides the BATS encoder with the following information:

- o Batch size (M): the number of coded packets in a batch.
- o Recoding field size (q): the number of elements in the finite field for recoding.
- o MAX_DEG: the size of DD.
- o The degree distribution (DD), which is an unsigned integer array of size MAX_DEG+1.
- o A sequence of batch IDs (j, j = 0, 1, ...).
- o Number of source packets (K).
- o Packet size (T): the number of octets in a source packet.
- o The source packets (b[i], i = 0, 1, ..., K-1).

Using this information, the (outer code) encoder generates a batch for each batch ID. For the batch ID j, the encoder returns the DDP that contains

- o a sparse degree d[j], and
- o M coded packets (x[j,i], i =0, 1, ..., M-1), each containing TO octets.

The DDP will use the batches to form DDP packets to be transmitted to other network nodes towards the destination nodes. The DDP MUST deliver with each coded packet its

- o d: sparse degree

- BID: batch ID

The DDP MUST deliver the following information to each recoder:

- M: batch size M

- q: recoding field size

The DDP MUST deliver the following information to each decoder:

- M: batch size

- q: recoding field size

- K: the number of source packets

- T: the number of octets in a source packet

The BID is used by both recoders and decoders. The BATS payload size TO MUST be known by all the nodes.

The DDP will also include some necessary extra information in the packet header so that the network nodes can identify different BATS sessions, and different end-to-end communication flows. However, such specifications are beyond the scope of this document.

2.2.3. Recoding Procedures

Both the source node and the intermediate nodes perform recoding on the batches before transmission. At the source node, the recoder receives the batches from the outer code encoding procedure. At an intermediate node, the DDP receives the DDP packets from the other network nodes, and should be able to extract coded packets and the corresponding batch information from these packets.

The DDP provides the recoder with the following information:

- the batch size M,

- the recoding field size q,

- a number of received coded packets of the same batch, each containing TO octets, and

- link statistics, e.g., packet loss rates.

For a received batch, the recoder determines a positive integer Mr, the number of recoded packets to be transmitted for the batch. The

recoder uses the information provided by the DDP to generate Mr recoded packets, each containing TO octets. The DDP uses the Mr recoded packets to form the DDP packets for transmitting.

2.2.4. Destination Node Procedures

A destination node needs the data transmitted by the source node. At the destination node, the DDP receives DDP packets from the other network nodes, and should be able to extract coded packets and the corresponding batch information from these packets.

The DDP provides the decoder with the following information:

- o M: batch size,

- o q: recoding field size,

- o K: the number of source packets

- o T: the number of octets of a source packet

- o A sequence of batches, each of which is formed by a number of coded packets belonging to the same batch, with their corresponding batch IDs and degrees.

The decoder uses this information to decode the K source packets. If successful, the decoder returns the recovered K source packets to the DDP, which will use the K source packets to form the F octets data. The recommended padding process is shown as follows:

```
// this procedure returns the number P of padding octets
// at the end of b[K-1]
Let bl be the last decoded source packet b[K-1]
PL = bl[T-1]
if PL == 1 do
    return P = 1
WI = T - 1
while bl[WI] == PL do
    WI = WI - 1
return P = (1 + bl[WI]) * bl[WI] + T - WI - 1
```

Figure 2: Data Depadding Process

2.3. Recommendation for the Parameters

The recommendation for the parameters M and q is shown as follows:

- o When q=2, M=16,32,64

- o When q=256, M=8,16,32,64

It is RECOMMENDED that K is at least 128. However, the encoder/ decoder SHALL support an arbitrary positive integer value less than 2^16.

2.4. Example DDP Packet Format

A DDP can form a DDP packet with a header (5 octets), a footer (3 octets) and a payload (TO octets). A DDP packet has totally 8+TO octets.

2.4.1. Packet Header

The BATS packet header has 40 bits (5 octets) and includes fields Packet_Count, Mq, Batch_ID, and Degree.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           Packet_Count        |  Mq   |       Batch_ID        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Degree    |
+-+-+-+-+-+-+-+-+
```

Figure 3: BATS packet header format.

- o Packet_Count: 16-bit unsigned integer, specifying the number K of packets of the BATS session.

- o Mq: 4-bit unsigned integer to specify the value of M and q as Table 1.

- o Batch_ID: 12-bit unsigned integer, specifying the batch ID BID of the batch the packet belonging to.

- o Degree: 8-bit unsigned integer, specifying the batch degree d of the batch the packet belonging to.

Mq	M	q	O
0010	16	2	2
0100	32	2	4
0110	64	2	8
0001	8	256	8
0011	16	256	16
0101	32	256	32
0111	64	256	64

Table 1: Values of Mq field

2.4.2. Packet Payload

```
              O                          T
+-----------------------+------------------------------+
|   coefficient vector  |          coded data          |
+-----------------------+------------------------------+
```

Figure 4: BATS packet payload format.

The payload has TO octets, where the first O octets contain the coefficient vector and the remaining T octets contain the coded data. Information in both fields MAY be encoded in JSON (ASCII) or protobuf (binary) formats.

o coefficient vector: O octets. The range of the value of O is in Table 1.

o coded data: T octets. T is at most MTU - 10, where 10 is the total of the header and footer length plus the minimum value of O.

2.4.3. Packet Footer

```
0                   1                   2
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|            signature          |  parity check |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 5: BATS packet footer format.

The footer has three octets.

o signature: 2 octets. A signature of the individual packet to prevent pollution attack.

o parity check: 1 octet. A parity check field used to verity the correctness of the packet.

3. BATS Code Specification

3.1. Common Parts

The T octets of a source packets are treated as a column vector of T elements in GF(256). Linear algebra and matrix operations over finite fields are assumed in this section.

Suppose that a pseudorandom number generator Rand() which generates an unsigned integer of 32 bits is shared by both encoding and decoding. The pseudorandom generator can be initialized by Rand_Init(S) with seed S. When S is not provided, the pseudorandom generator is initialized arbitrarily. One example of such a pseudorandom generator is defined in RFC 8682 [RFC8682].

A function called BatchSampler is used in both encoding and decoding. The function takes two integers j and d as input, and generates an array idx of d integers and a d x M matrix G. The function first initializes the pseudorandom generator with j, sample d distinct integers from 0 to K-1 as idx, and sample d*M integers from 0 to 255 as G. See the pseudocode in Figure 6.

```
function BatchSampler(j,d)
    // initialize the pseudorandom generator by seed j.
    Rand_Init(j)
    // sample d distinct integers between 0 and K-1.
    for k = 0, ..., d-1 do
        r = Rand() % K
        while r already exists in idx do
            r = Rand() % K
        idx[k] = r

    // sample d x M matrix
    for r = 0, ..., d-1 do
        for c = 0,...,M-1 do
            G[r,c] = Rand() % 256

    return idx, G
```

Figure 6: Batch Sampler Function

3.2. Outer Code Encoder

Define a function called DegreeSampler that return an integer d using the degree distribution DD. We expect that the empirical distribution of the returning d converges to DD(d) when d < K. One design of DegreeSampler is illustrated in Figure 7.

```
function DegreeSampler(j, DD)
    Let CDF be an array
    CDF[0] = 0
    for i = 1, ..., MAX_DEG do
        CDF[i] = CDF[i-1] + DD[i]
    Rand_Init()
    r = Rand() % CDF[MAX_DEG]
    for d = 1, ..., MAX_DEG do
        if r >= CDF[d] do
            return min(d,K)
    return min(MAX_DEG,K)
```

Figure 7: Degree Sampler Function

Let b[0], b[1], ..., b[K-1] be the K source packets. A batch with BID j is generated using the following steps.

- o Obtain a degree d by calling DegreeSampler with input j.

- o Obtain idx and G[j] by calling BatchSampler with input j and d.

- o Let B[j] = (b[idx[0]], b[idx[1]], ..., b[idx[d-1]]). Form the batch X[j] = B[j]*G[j], whose dimension is T x M.

- o Form the TO x M matrix Xr[j], where the first O rows of Xr[j] form the M x M identity matrix I with entries in GF(q), and the last T rows of Xr[j] is X[j].

See the pseudocode of the batch generating process in Figure 8.

```
function GenBatch(j)
    d = DegreeSampler(j)
    (idx, G) = BatchSampler(j,d)
    B = (b[idx[0]], b[idx[i]], ..., b[idx[d-1]])
    X = B * G
    Xr = [I_M; X]
    return Xr
```

Figure 8: Batch Generation Function

3.3. Inner Code Encoder (Recoder)

The inner code comprises (random) linear network coding applied on the coded packets belonging to the same batch. At a particular network node, recoded packets are generated by (random) linear combinations of the received coded packets of a batch. The recoded packets have the same BID, sparse degree and coded packet length.

The number Mr of recoded packets for a batch is decided first by the recoder. Mr can be set as M. When the link statistics is known, the recoder can try to obtain the link packet loss rate e for the link to transmit the recoded batch, and set Mr to be (1+e)M.

Suppose that coded packets xr[i], i = 0, 1, ..., r-1, which have the same BID j, have been received at an intermediate node. Using the recommended packet format, it can be verified whether the corresponding packet headers of these coded packets are the same. Then a recoded packet can be generated by one of the following two approaches:

- forwarding: when receiving xr[i], directly use xr[i] as a recoded packet.

- linear combination recoding: (randomly) choose a sequence of coefficients c[i], i = 0, 1, ..., r-1 from GF(q). Generate c[0]xr[0]+c[1]xr[1]+...+c[r-1]xr[r-1] as a recoded packet.

A recoder can combine these two approaches to generate recoded packets. For example, the recoder will output xr[i], i = 0, 1, ..., r-1 as r systematic recoded packets and generate Mr-r recoded packets using linear combinations of randomly chosen coefficients.

3.4. Belief Propagation Decoder

The decoder receives a sequence of batches Yr[j], j = 0, 1, ..., n-1, each of which is a TO-row matrix over GF(256). The degree d[j] of batch j is also known. Let Y[j] be the submatrix of the last T rows of Yr[j]. When q = 256, let H[j] be the first M rows of Yr[j]; when q = 2, let H[j] be the matrix over GF(256) formed by embedding each bit in the first M/8 rows of Yr[j] into GF(256).

By calling BatchSampler with input j and d[j], we obtain idx[j] and G[j]. According to the encoding and recoding processes described in Section 3.2 and Section 3.3, we have the system of linear equations Y[j] = B[j]G[j]H[j] for each received batch with ID j, where B[j] = (b[idx[j,0]], b[idx[j,1]], ..., b[idx[j,d-1]]) is unknown.

We describe a belief propagation (BP) decoder that can efficiently solve the source packets when a sufficient number of batches have been received. A batch j is said to be decodable if rank(G[j]H[j]) = d[j] (i.e., the system of linear equations Y[j] = B[j]G[j]H[j] with B[j] as the variable matrix has a unique solution). The BP decoding algorithm has multiple iterations. Each iteration is formed by the following steps:

- o Decoding step: Find a batches j that is decodable. Solve the corresponding system of linear equations Y[j] = B[j]G[j]H[j] and decode B[j].

- o Substitution step: Substitute the decoded source packets into undecodable batches. Suppose that a decoded source packet b[k] is used in generating a undecodable Y[j]. The substitution involves 1) removing the entry in idx[j] corresponding to k, 2) removing the row in G[j] corresponding to b[k], and 3) reducing d[j] by 1.

The BP decoder repeats the above steps until no batches are decodable during the decoding step.

4. Research Issues

The baseline BATS coding scheme described in Section 2 and Section 3 needs various refinement and complement towards a more sophisticated network communication application. Various related research issues are discussed in this section, but the security related issues are left to Section 6.

4.1. Coding Design Issues

The BATS code specification in Section 3 has nearly optimal throughput when the number of source packets K is sufficiently large. But when K is small, the degree sampler function in Figure 7 and the BatchSampler function in Figure 6 based on a pseudorandom generator may not sample all the source packets evenly, so that some of the source packets are not well protected. One approach to solve this issue is to generate a deterministic degree sequence when the number of batches is relatively small, and design a special pseudorandom generator that has a good sampling performance when K is small.

The belief propagation decoder in Section 3.4 guarantees the recovery of a given fraction of the source packets. To recover all the source packets, a precode can be applied to the source packets to generate a fraction of redundant packets before applying the outer code encoding. Moreover, when the belief propagation decoder stops, it is possible to continue with inactivation decoding, where certain source packets are treated inactive so that a similar belief propagation

process can be resumed. The reader is referred to RFC 6330 [RFC6330] for the design of a precode with a good inactivation decoding performance.

There are research issues related to recoding discussed in Section 3.3. One question is how many recoded packets to generate for each batch. Though it is asymptotically optimal when using the same number of recoded packets for all batches, it has been shown that transmitting a different number of recoded packets for different batches can improve the recoding efficiency. The intuition is that for a batch with a lower rank, a smaller number of recoded packets need to be transmitted. This kind of recoding scheme is called adaptive recoding [Yin19].

Packet loss in network communication is usually bursty, which may harm the recoding performance. One way to resolve this issue is to transmit the packets of different batches in a mixed order, which is also called batch interleaving [Yin20]. How to efficiently interleave batches without increasing too much end-to-end latency is a research issue.

Though we only focus on the BATS coding scheme with one source node and one destination node, a BATS coding scheme can be used for multiple source and destination nodes. To benefit from multiple source nodes, we would need different source nodes to generate statistically independent batches. For communicating the same data to multiple destination nodes, which is also call multicast, it is well-known that linear network coding [Li03] achieves the mulicast capacity. BATS codes can benefit from network coding due to its inner code, but how to efficiently implement multicast needs further research.

4.2. Protocol Design Issues

The baseline scheme in this document focuses on the reliable communication. There are other issues to be considered towards designing a fully functionally DDP based on a BATS coding scheme. Here we discuss some network management issues that are closely related to a BATS coding scheme: routing, congestion control and media access control.

The outer code of a BATS code can be regarded as a channel code for the channel induced by the inner code, and hence the network management algorithms should try to maximize the capacity of the channel induced by the inner code. A network utility maximization problem [Dong20] for BATS coding can be applied to study routing, congestion control and media access control jointly. Compared with the network utility maximization for Internet, there are two major

differences. First, the network flow rate is not measured by the rate of the raw packets. Instead, a rank based measurement induced by the inner code is applied for BATS coding schemes. Second, due to recoding, the raw packet rate of a flow may not be the same for different links, i.e., no flow conservation for BATS coding schemes. These differences affect both the objective and the constraints of the utility maximization problem.

Practical congestion control, routing and media access control algorithms for BATS coding schemes deserve more research efforts. Due to the recoding operation, congestion control cannot be only performed end-to-end. The rate of transmitting batches can be controlled end-to-end, but the number of recoded packets generated for a batch must be controlled at the intermediate nodes, which introduces new research issues for congestion control. For routing, the BATS coding scheme is flexible for implementing multi-path data transmission, and different batches can be transmitted on a different path between a source node and a destination node. Under the scenario of BATS coding schemes, media access control can have some different considerations: Retransmission is not necessary, and a reasonably high packet loss rate can be tolerated.

4.3. **Application Related Issues**

There are more researche issues pertaining to different applications. The reliable communication technique provided by BATS codes can be used for a broad range of network communication scenarios. In general, a BATS coding scheme is suitable for data delivery in networks with multiple hops and unreliable links.

One class of typical application scenario is wireless mesh and ad hoc networks [Toh02], including vehicular networks, wireless sensor networks, smart-lamppost networks, etc. These networks are characterized by a large number of network devices connected wirelessly with each other without a centralized network infrastructure. A BATS coding scheme is suitable for high data load delivery in such networks without the requirement that the point-to-point/one-hop communication is highly reliable. Therefore, employing a BATS coding scheme can provide more freedom for media access control, including power control so that the overall network throughput can be improved.

Another typical application scenario of BATS coding schemes is underwater acoustic networks [Sprea19], where the propagation delay of acoustic waves in underwater can be as long as several seconds. Due to the long delay, feedback based mechanisms become inefficient. Moreover, point-to-point/one-hop underwater acoustic communication (for both the forward and reverse directions) is highly unreliable.

Due to these reasons, traditional networking techinques developed for radio and wireline networks cannot be directly applied to underwater networks. As a BATS coding scheme does not rely on the feedback for reliability communication and can tolerate highly unreliable links, it makes a good candidate for developing data delivery protocols for underwater acoustic networks.

Last but not least, due to its capability of performing multi-source, multi-destination communications, a BATS coding scheme can be applied in various content distribution scenarios. For example, a BATS coding scheme can be a candidate for the erasure code used in the liquid data networking framework [Byers20] of CCN (content centric networking), and provides the extra benefit of network coding [Zhang16].

5. IANA Considerations

This memo includes no request to IANA.

6. Security Considerations

Subsuming both Random Linear Network Codes (RLNC) and fountain codes, BATS codes naturally inherit both their desirable capability of offering confidentiality protection as well as their vulnerability towards pollution attacks.

6.1. Provision of Confidentiality Protection

Since the transported messages are linearly combined with random coefficients at each recoding node, it is statistically impossible to recover the individual messages by capturing the coded messages at any one or small number of nodes. As long as the coding matrices of the transported messages cannot be fully recovered, any attempt of decoding any particular symbol of the transported messages is equivalent to random guessing [Bhattad05].

The threat towards confidentiality, however, also exists in the form of eavesdropping on the initial encoding process, which takes place at the encoding nodes. In these nodes, the transported data are presented in plain text and can be read along their transfer paths. Hence, information isolation between the encoding process and all other user processes running on the node must be assured.

In addition, the authenticity and trustworthiness of the encoding, recoding and decoding program running on all the nodes must be attested by a trusted authority. Such a measure is also necessary in countering pollution attacks.

6.2. Countermeasures against Pollution Attacks

Like all network codes, BATS codes are vulnerable under pollution attacks. In these attacks, one or more compromised coding node(s) can pollute the coded messages by injecting forged messages into the coding network and thus prevent the receivers from recovering the transported data correctly.

The research community has long been investigating the use of various signature schemes (including homomorphic signatures) to identify the forged messages and stall the attacks (see [Zhao07], [Yu08], [Agrawal09]). However, these countermeasures are regarded as being too computationally expensive to be employed in broadband communications. Hence, a system-level approach based on Trusted Computing [TC-Wikipedia] is proposed as a practical alternative to protect BATS codes against pollution attacks. This Trusted Computing based protection consists of the following countermeasures:

1. Attestation and Validation of all BATS encoding, recoding and decoding nodes in the network. Remote attestation and repetitive validation of the identity and capability of these node based on valid public key certificates with proper authorization MUST be a pre-requisite for admitting these nodes to a network and permitting them to remain on that network.

2. Attestation of all encoding, recoding and decoding programs used in the coding nodes. All programs used to perform the BATS encoding, recoding and decoding processes MUST be remotely attested before they are permitted to run on any of the coding nodes. Reloading or alteration of programs MUST NOT be permitted during an encoding session. Programs MUST be attested or validated again when they are executed in new execution environments instantiated even in the same node.

3. Original Authentication of all coded messages using network level security protocols such as IPsec or Peer Authentication over session-based communication using transport level security protocols such as TLS/DTLS MUST be employed in order to provide Message Origin or Communication Peer Authentication to every coded message sent through the coding network.

7. References

7.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC8682] Saito, M., Matsumoto, M., Roca, V., Ed., and E. Baccelli, "TinyMT32 Pseudorandom Number Generator (PRNG)", RFC 8682, DOI 10.17487/RFC8682, January 2020, <https://www.rfc-editor.org/info/rfc8682>.

7.2. Informative References

[Agrawal09] Agrawal, S. and D. Boneh, "Homomorphic MACs: MAC-based integrity for network coding", International Conference on Applied Cryptography and Network Security , 2009.

[Bhattad05] Bhattad, K. and K. Narayanan, "Weakly Secure Network Coding", ISIT , 2007.

[Byers20] Byers, J. and M. Luby, "Liquid Data Networking", ICN , 2020.

[Dong20] Dong, Y., Jin, S., Yang, S., and H. Yin, "Network Utility Maximization for BATS Code enabled Multihop Wireless Networks", ICC , 2020.

[Li03] Li, S., Yeung, R., and N. Cai, "Linear Network Coding", IEEE Transactions on Information Theory , 2003.

[RFC6330] Luby, M., Shokrollahi, A., Watson, M., Stockhammer, T., and L. Minder, "RaptorQ Forward Error Correction Scheme for Object Delivery", RFC 6330, DOI 10.17487/RFC6330, August 2011, <https://www.rfc-editor.org/info/rfc6330>.

[Sprea19] Sprea, N., Bashir, M., Truhachev, D., Srinivas, K., Schlegel, C., and C. Claudio Sacchi, "BATS Coding for Underwater Acoustic Communication Networks", OCEANS , 2019.

[TC-Wikipedia] "Trusted Computing", Wikipedia https://en.wikipedia.org/wiki/Trusted_Computing.

[Toh02] Toh, C., "Ad Hoc Mobile Wireless Networks", Prentice Hall Publishers , 2002.

[Yang14] Yang, S. and R. Yeung, "Batched Sparse Codes", IEEE Transactions on Information Theory 60(9), 5322-5346, 2014.

[Yang17] Yang, S. and R. Yeung, "BATS Codes: Theory and Practice", Morgan & Claypool Publishers , 2017.

[Yin19] Yin, H., Tang, B., Ng, K., Yang, S., Wang, X., and Q. Zhou, "A Unified Adaptive Recoding Framework for Batched Network Coding", ISIT , 2019.

[Yin20] Yin, H., Yeung, R., and S. Yang, "A Protocol Design Paradigm for Batched Sparse Codes", Entroy , 2020.

[Yu08] Yu, Z., Wei, Y., Ramkumar, B., and Y. Guan, "An Efficient Signature-Based Scheme for Securing Network Coding Against Pollution Attacks", INFOCOM , 2008.

[Zhang16] Zhang, G. and Z. Xu, "Combing CCN with network coding: An architectural perspective", Computer Networks , 2016.

[Zhao07] Zhao, F., Kalker, T., Medard, M., and K. Han, "Signatures for content distribution with network coding", ISIT , 2007.

Appendix A. Additional Stuff

Authors' Addresses

Shenghao Yang
CUHK(SZ)
Shenzhen, Guangdong
China

Phone: +86 755 8427 3827
Email: shyang@cuhk.edu.cn

Xuan Huang
CUHK
Hong Kong, Hong Kong SAR
China

Phone: +852 3943 8375
Email: 1155136647@link.cuhk.edu.hk

Raymond W. Yeung
CUHK
Hong Kong, Hong Kong SAR
China

Phone: +852 3943 8375
Email: whyeung@ie.cuhk.edu.hk

John K. Zao
NCTU
Hsinchu, Taiwan
China

Email: jkzao@ieee.org