Privacy and Confidentiality in Content-Based Networking

Abdullatif Shikfa
EURECOM
France
Email: Abdullatif.Shikfa@eurecom.fr

Melek Önen
EURECOM
France
Email: Melek.Onen@eurecom.fr

***Abstract*—Privacy and confidentiality are crucial issues in content-based networking. In this paper, we present security primitives required to achieve privacy in content-based networks. We define three privacy models adapted to content-based networking and detail what are the requirements that the security primitives have to achieve in order to fit in each of these models. We also propose an original protocol that features full privacy content-based networking.**

I. Introduction

Opportunistic networks are based on a novel communication paradigm that aims at overcoming the limitations of communication services built upon the widely used concept of end-to-end connectivity. Indeed, users have nowadays "islands of end-to-end connectivity" at home, at the office or in hotspots. However, they are also likely to sporadically be in range of many other users while in between and, in spite of enjoying ever increased connectivity, they cannot benefit of end-to-end communications over several different technologies at a time.

Opportunistic and autonomic networking is designed to solve the problem of communication in the presence of intermittent network connectivity, and, to this end, has the following requirements:

- **relaxed end-to-end connectivity:** opportunistic networking aims at transmitting a message over any communication medium available. To achieve such a goal, message routing has to be very dynamic, forwarding decisions are taken on-the-fly so that packets eventually reach their destination but establishing an end-to-end path is not envisageable.
- **collapsed architecture:** in order to benefit from various communication architectures, packets created to take advantage of opportunistic networking have a collapsed architecture where all information whether concerning the application or networking operations is at the same level. With such a cross-layer design packets can be slightly modified to fit any network they are forwarded through.

A concept that nicely fits with the underlying opportunistic networking model is offered by content-based communication ([3], [4]) whereby messages are forwarded from source to destinations based on their content rather than explicit addresses. In a content-based communication service, receivers declare their interests through receiver advertisements while senders simply publish messages without specifying a destination. Receiver adverisements work then as selection predicates over the published content. Message content is namely structured as a set of attribute/value pairs and advertisements act as constraints over attributes. For example the message:

$$[type = music; style = classical; composer = Beethoven; \\ title = 9^{th} symphony; data = bytearray]$$

would match an advertisement such as

$$[type = music; style = classical].$$

Privacy and confidentiality are crucial issues in content-based networking. Advertisements and published content are namely forwarded through various intermediate nodes that may not be trusted by sources or receivers ; moreover, trust relationship are loose in such a heterogenous environment. Receivers do not want any other node (especially untrusted ones) to know what their interests are because these information threaten their privacy. Thus, nodes should be able to correctly build their forwarding tables based on encrypted advertisements and they further should correctly forward encrypted content based on these forwarding tables. Hence, nodes require mechanisms that allows to take content-based forwarding decisions without accessing the content in clear. In [5], Lilien et al. present the challenges in privacy and security of opportunistic networks but, to the best of our knowledge, we are the first to study the problems of privacy and confidentiality in content-based networks. The main contributions of this paper are the following:

- We present security primitives required to achieve privacy in content-based networks. We define three privacy models adapted to content-based networking and detail what are the requirements that the security primitives have to achieve in order to fit in each of these models,
- We propose an original protocol that features complete content-based networking with strong privacy enforcement.

In the next section, we first illustrate the problem of privacy in content-based networking with an example, and derive from it two main security primitives. In section III, we formally define three privacy models and then detail them regarding the two security primitives. Section IV analyzes two basic approaches that attempt to solve the problem of privacy

enforcement in content based networking whereas section V presents our original scheme.

II. Problem statement

A. An example

In order to formally define the different security requirements in content based networks, we first propose to illustrate such communications with the following example: a music sharing application in a vehicular network where Alice and Amanda are two receivers, Bob is an intermediate node whose task is to forward messages, and Charlie is a publisher. In this typical example, there are many privacy covers that should be taken into account. For example, as an intermediate node, Bob may not have the right to access the content of packets that he is forwarding. The whole scenario is schematized in figure 1. Let us then consider that Alice is stuck in a traffic jam. In order to enjoy her time she would like to listen to some music. Thus, she sends an advertisement to the cars around her requesting some classical music, as follows:

$$[type = music; style = classical].$$

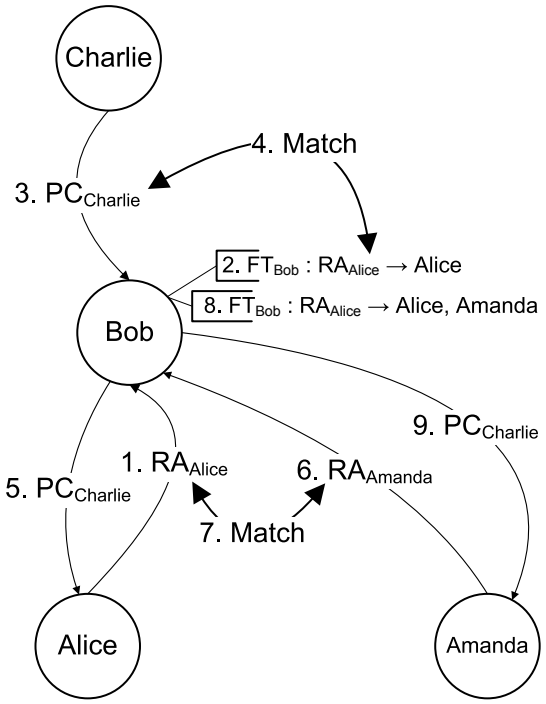

Fig. 1. Scenario schema

Since she doesn't want anybody to know what she is requesting, she is going to encrypt her request in some way, with an appropriate algorithm that we will denote by E_{Alice}. Thus, she will broadcast a modified message:

$$RA_{Alice} = E_{Alice}([type = music; style = classical]).$$

When Bob who is driving a car in the transmission range of Alice, receives Alice's request, he should not be able to decipher it. He may though forward the packet and add a new entry to his forwarding table FT_{Bob} as such:

$$RA_{Alice} \rightarrow Alice.$$

He thus forwards Alice's advertisement to other nodes using its own transmission interface. Let us now consider a third user Charlie, who wishes to share his music with other people stuck in the traffic jam. He then broadcasts the files he wishes to share through well-formated messages such as follows:

$$\begin{aligned} PC_{Charlie} \quad = \quad & [type = music; style = classical; \\ & composer = Beethoven; title = 9^{th} symphony; \\ & data = bytearray] \end{aligned}$$

where the field data is the actual music, the payload. Since Charlie also has confidentiality concerns, he also encrypts his message with an appropriate algorithm that we will denote by $E'_{Charlie}$, therefore he sends:

$$\begin{aligned} PC_{Charlie} \quad = \quad & E'_{Charlie}([type = music; style = classical; \\ & composer = Beethoven; title = 9^{th} symphony; \\ & data = bytearray]) \end{aligned}$$

The difference between E and E' is that the first one only takes receiver advertisements into account (that is to say control messages) whereas the other also encrypts some payload. Thus E and E' may be the same algorithm but it is more likely that there will be two different algorithms and that is why we use two distinct notations. This message is then forwarded through the network and at sometime it arrives to Bob. Although the message is encrypted, Bob should be able to perform a look-up with its forwarding table and detect that this content matches Alice's advertisement. He only learns that this message fits Alice's request, without discovering what Alice's request is. He thus forwards the message to Alice who can happily enjoy her music.

Let us now consider one more case here, that is the case where two or more users share the same interest. Indeed, imagine that when Bob receives the request from Alice, he also receives another request from another user Amanda, and her request is also encrypted. Amanda is looking for the same content, her request is as follows:

$$RA_{Amanda} = E_{Amanda}([type = music; style = classical]).$$

At this point, it is important that Bob manages to identify that Amanda's and Alice's advertisements are the same even though their encrypted form is different. So Bob should have a way to update his forwarding table FT_{Bob} such as follows:

$$(RA_{Alice} \Leftrightarrow RA_{Amanda}) \rightarrow (Alice, Amanda).$$

This aggregation of requests allows to build an efficient forwarding table, and therefore optimizes the memory overhead.

B. Security primitives

As we have seen with the example in the previous section, security has to be enforced with several operations. First, receiver advertisements have to be encrypted to enforce privacy. However, encrypted advertisements should be forwarded towards the network and intermediate nodes should be able to build forwarding tables based on these encrypted information. Therefore such applications require a dedicated encryption operation that allows some networking operations over encrypted data. However, in order to optimize bandwith usage, similar advertisements should first be aggregated and forwarded into a single packet. Therefore, intermediate nodes should be able to first compare encrypted packets and aggregate them into one packet if they are equivalent and finally forward this single packet.

As for receiver advertisements, a content publisher may require some security operations in order not to let unauthorized nodes access the content. In this case, the encryption operation performed by the publisher should also provide some similar properties as the one for advertisements. Indeed, encrypted content should correctly be forwarded by intermediate nodes and should finally reach its corresponding recipients. Therefore, whenever a content is received, whether it is encrypted or not, an intermediate node should be able to take a forwarding decision over this content based on its forwarding table.

To summarize, forwarding decisions are directly taken over the content of the packet but content publishers or receivers may not wish to reveal this content to some intermediate nodes whose only task is forwarding. In order to ensure both networking and security, intermediate nodes require two main security primitives:

- **secure building of forwarding tables**: in order to correctly forward packets, intermediate nodes must construct a forwarding table based on recipients advertisements, whether they are encrypted or not ;
- **secure look-up**: based on this forwarding table, an intermediate node must be able to take some correct forwarding decision whenever it receives a content. This content may also sometimes be encrypted.

The design of these two security primitives can differ with respect to the application security requirements and mainly with respect to the level of privacy. The mechanisms required to achieve a certain level of privacy inherently depend on the level of trust between intermediate nodes on the one hand, and receivers or content publishers on the other hand. Indeed, if an intermediate node is totally trusted for example, then it may have the right to access the content of the data in order to take forwarding decisions.

In the example described in the previous section, nodes do not trust the intermediate node B and therefore B cannot have access to the content of the packets. In this example, users want to achieve the strongest privacy in the worst environment (no trust at all between nodes). This is kind of an ultimate scenario but there are intermediate cases of course. In the next section, we are going to define several privacy models and their application to content-based forwarding.

III. Security definitions

A. Privacy models

As explained in the previous section, the design of the two security primitives depends on the level of privacy required from the application. A content publisher or a receiver may or may not want to reveal some content or some interests respectively to the intermediate nodes. After analyzing many different scenarios in content based applications, we came up with three privacy models:

- **model 1, privacy oblivious**: this model refers to the case where publishers or receivers do not require security at all. Therefore, information is simply sent in clear and intermediate nodes proceed as in standard content-based applications.
- **model 2, intra-community privacy**: in this model, the level of privacy depends on node's relationship. Indeed, some intermediate nodes may be trusted and some others not. The trust relationship can for example be based on some community membership. In this case, members of a community agree on sharing information with other members of the same community. Therefore, intermediate nodes can decrypt some packets if they belong to the same community of a receiver or a publisher.
- **model 3, full privacy**: as opposed to model 2, this model refers to the case where nodes do not trust any other node. Therefore, intermediate nodes should be able to process some encrypted packets without having access to the content of these packets. In this model, any node becomes a potential attacker.

In the next section, we discuss the design of the two security primitives that are secure building of forwarding tables and secure look-up based on these three privacy models.

B. Privacy-aware building of forwarding tables and look-up

As decribed in section II, some nodes denoted by A_i send their interest as receiver advertisements RA_i towards the network. Other nodes denoted by C_j and considered as being publishers, send some content PC_j. Intermediate nodes, denoted by B_k are therefore in charge of forwarding both advertisements and published content. This published content is composed of two parts: control information and the payload itself. Only the control information, that we denote by CI_j are relevant for the look-up operation. It is worth noting that this classification of nodes in three categories is purely functional: it is indeed possible that one node assumes the three roles of receiver, publisher and intermediate node depending on the communication that is undergoing.

In order to correctly and efficiently perform network operations, B_k first needs to build a forwarding table, denoted by FT_k based on the received RA_i and further uses this table to take forwarding decisions whenever it receives a PC_j. The design of these two primitives strongly depends on the privacy models described in the previous section. We therefore analyze these two problems for each of these models.

- **model 1, privacy oblivious**: when no privacy is required at all, both RA_i and CI_j are received by intermediate nodes in clear. In this case, the building of forwarding tables FT_k and the look-up operations are the classical ones used in content based networking. Therefore, whenever B_k receives a RA_i, it first looks if such an entry or an equivalent one exists in its forwarding table. If this is not the case, then B_k adds an additional row in its table as follows:

  $$RA_i \rightarrow A_i.$$

  If, on the other hand, B_k finds an equivalence between the received RA_i and another one in its forwarding table denoted by $RA_{i'}$ then B_k aggregates this information and updates the row corresponding to $RA_{i'}$ as follows:

  $$(RA_i \Leftrightarrow RA_{i'}) \rightarrow A_{i'}, A_i.$$

  Once the forwarding table FT_k is built, B_k can propagate the aggregated advertisement towards the network and correctly make forwarding decisions whenever it receives a packet PC_j. Indeed, the look-up operation consists in comparing the control information CI_j of PC_j with each row in its forwarding table in order to define the next hop for the packet. This case with no privacy can be used as a witness case.
- **model 2, intra-community privacy**: in this model, recipients and publishers only trust B_k if they belong to the same community. In this case, B_k is able to decrypt any packet originating from members of its community. For example, suppose that A_1 and B_1 belong to one community ($community1$) and A_2 and B_1 to another community ($community2$). If A_1 and A_2 send their common interest to B_1, only B_1 will be able to discover that RA_1 and RA_2 are equivalent and therefore update its forwarding table (but neither eavesdroppers nor A_1 or A_2 would be able to detect this equivalence). Similarly, when C_1 sends some encrypted data to B_k, if B_k belongs to the same community, then it can have access to the control information CI_1 and perform a correct look-up without revealing any extra information to potential eavesdroppers.
- **model 3, full privacy**: in this model, every node becomes a potential adversary. This implies that A_i or C_j do not trust any inermediate node B_k and therefore they encrypt their advertisements or content packet respectively. To build its forwarding table, B_k should first be able to detect whenever two encrypted advertisements RA_i and $RA_{i'}$ are equivalent without decrypting them as opposed to the case in model 2. The only information that it should get from this process is the matching between them, it should never be able to get more information on the interests. Similarly, the content publisher will encrypt its packet PC_j and B_k should be able to find whether the encrypted control information CI_j within this packet matches one of the encrypted entries of its forwarding table FT_k or not. Therefore, B_k will always know where to forward the packet without knowing neither the content of the message nor the corresponding advertisement.

Now that we have clearly defined the privacy models for each security primitive, we analyze some basic approaches to solve these problems, and then propose a complete privacy solution.

IV. Basic approaches and their drawbacks

A. Hash functions

The first basic idea to solve these problems is to use a cryptographic hash function, as proposed by Propicman in [6]. A cryptographic hash function is a one-way collision resistant function. Receivers (A_1 and A_2) hash their advertisements using a public hash function h and send them to the intermediate node B. B receives $h(RA_1)$ and $h(RA_2)$, compares them and if they are the same, he puts them in the same row of its forwarding table, otherwise he puts them in two different rows. B is therefore able to detect if they are equivalent or not without learning their actual value (because by their very definition, finding x given $h(x)$ is difficult). When C wants to send a message, he also performs a hash function over the control information before sending it. B receives $h(CI)$ from C and he has to do a look up in his forwarding table. He can do it on hashed values directly since if $CI = RA_1$ for example, then $h(CI) = h(RA_1)$. B can then perform the secure look-up and forward the message as indicated by its forwarding table without accessing the hidden information.

The idea looks seducing and efficient, it almost achieves model 3 of privacy and its cost is very low. Yet, it presents a major flaw called the dictionary attack. Since the hash function is public and no secret is required, any node, including A_1, A_2, B, C or an attacker, can compute the hash of any value. Since the messages are well formated and they have a meaning (which is very different from a pseudo-random sequence), B or another attacker could simply compute the hashes of all words of a dictionary and then identify these hashes with the hashed value exchanged during the protocol. This attack is quite cheap and can easily and efficiently be launched thus breaking confidentiality of hashed values and impacting privacy. In fact this method does not even achieve model 2 because of this simple attack.

B. Group security

Another idea is to use group key cryptography in order to achieve intra-community privacy. The idea would be that nodes trusting each other share a common key, for example all nodes belonging to a given community are given a common key, that we call community key. For example, let us suppose that A_1, B and C belong to $community1$ and thus share key k_1, and that A_2 and B belong to $community2$ and thus share key k_2. Then, A_1 sends $E_{k_1}(RA_1)$ to B which can decipher it and access to RA_1 contrary to attackers which are not member of community 1. Similarly, A_2 sends $E_{k_2}(RA_2)$ to B who deciphers it and builds its forwarding table in cleartext. When C wants to send his message, he sends $E_{k_1}(CI)$ to B who can then decipher it and perform the

look-up operation in a classical way and forward the message afterwards. Eavesdroppers have no access to information since it is encrypted but members of the community have access to all information. For example, A_1 can directly decipher $E_{k_1}(CI)$ if she catches it, but this is normal since A_1 and C trust each other.

This mechanism fulfills goal of model 2 in terms of privacy, but it has some disadvantages. First of all, group key cryptography implies heavy key management to build the groups, add members or revoke some. Such an administrative burden should be taken care of and might not be available depending on the network capabilities. Another problem is that nodes which do not belong to a given community are completely excluded. For example, suppose that A_2 is not member of any community to which B belongs. Then, A_2 and B cannot use this mechanism and they need an alternative one. Even worse, suppose that, in order to communicate with B, C needs to send its messages through a node D that is trusted neither by B nor C. Then, C cannot use this protocol to send its message and the information would never reach B. This means that we cannot use this method only, we need to use in addition to it a model 3 method for the cases when nodes don't trust each other. So this method is not standalone, but it can be used in a hybrid protocol.

V. A PRELIMINARY SOLUTION: PRIVACY WITH MULTIPLE ENCRYPTION

A. General idea

We now present an original approach, which solves the problem of full privacy content-based forwarding. This approach is inspired by the mechanism of secure data aggregation with multiple encryption presented in [7]. In this scheme, advertisements are encrypted with symmetric keys by a simple XOR operation.

The scenario is as following: we have a tree network where all nodes (whether leaves or intermediate nodes) advertise their interest towards the path from these nodes to the root node, which is a content distributor. This scenario is illustrated in figure 2. As stated in the description of the privacy model 3, nodes do not trust any other node. They only trust the root node C but do not want to reveal any information to other nodes. So instead of simply advertising their interest w, they advertise it in an encrypted way. The encryption operation E that is used in our proposed scheme is a simple XOR with a pseudo-randomly generated key k. Therefore $E(w) = w \oplus k$. The confidentiality of these keywords is assured with the use of multiple encryption layers that are added and/or removed at each intermediate node. Thanks to this mechanism, no node except the root node can discover other nodes' interest even if they share the same interest.

B. Key distribution

The choice of the keys and hence the key distribution plays a crucial role in our solution. We suppose that each node shares two keys with each of its parent, grandparent, children and grandchildren when they exist. The first key is used in order to perform the two security primitives, and the second one is only used to ensure data confidentiality. The keys shared between two nodes N and M are denoted by k_{MN} (respectively k'_{MN}) or k_{NM} (respectively k'_{NM}) indifferently. In figure 2, B_1 shares two different keys with its children A_1, A_2, its parent B_3 and its grandparent B_4 denoted respectively by $k_{B_1A_1}$, $k_{B_1A_2}$, $k_{B_1B_3}$, $k_{B_1B_4}$ and $k'_{B_1A_1}$, $k'_{B_1A_2}$, $k'_{B_1B_3}$, $k'_{B_1B_4}$. In addition to these shared keys, intermediate nodes require an additional key defined as aggregation secret that is used for verifying advertisement's equivalence. This aggregation secret consists in the XOR of the shared keys between two children and the parent of the intermediate node. For example, node B_1 knows the aggregation secret for nodes A_1 and A_2 denoted by $as_{A_1A_2}$, which is equal to $as_{A_1A_2} = k_{A_1B_3} \oplus k_{A_2B_3}$. This aggregation secret is computed by B_3 and sent to B_1. In the next sections we describe how our solution deals with the problems of building secure forwarding table and secure look-up.

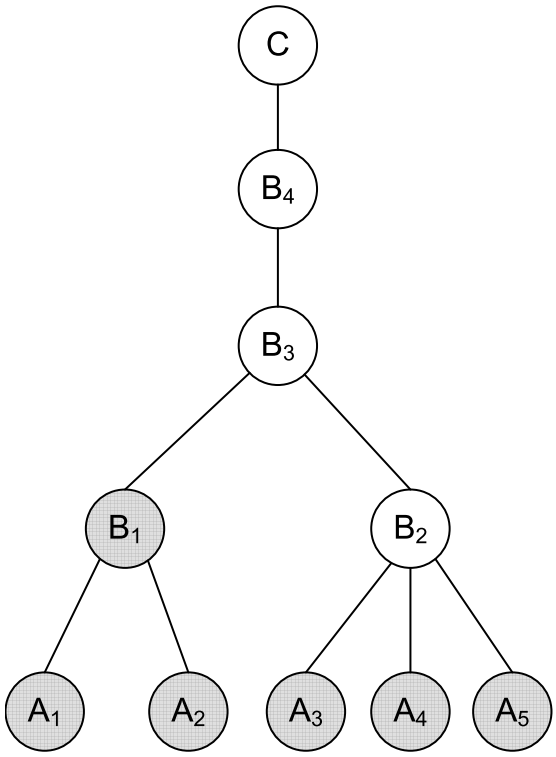

Fig. 2. Network used as illustration

C. Advertisements propagation and forwarding table building

We first study the case where two nodes of the same level share the same interest, then we describe the case where an intermediate node and its child share the same interest and finally we detail the case where more than two nodes of the same level share the same interest.

We suppose that nodes A_1 and A_2 want to advertise about a common interest in the word w (but they don't know that their interest is common). When a node sends its own

advertisement, it encrypts it with the key it shares with its parent and its grandparent and sends it to its parent. It also indicates that this advertisement is its own in a second part of the message. For example A_1 sends

$$[RA_{A_1} = w \oplus k_{B_1A_1} \oplus k_{B_3A_1}; A_1]$$

to B_1. When an intermediate node receives the advertisement of one of his children, the first step is to remove their shared key from the process. Indeed, the addition of this encryption layer corresponding to the key shared between a node and its parent ($k_{B_1A_1}$ for A_1 for example) has the purpose of preventing the grandparent from eavesdroping on the communication between a node and its parent and to be able to decipher the advertisement to get to w. For instance, when B_1 receives RA_{A_1} for A_1, it first computes

$$m_1 = RA_{A_1} \oplus k_{B_1A_1} = w \oplus k_{B_3A_1}.$$

This being done, B_1 cannot find what w is, but he adds to its forwarding table a row as follows:

$$w \oplus k_{B_3A_1} \rightarrow A_1.$$

Since A_2 is interested in the same word w, he acts as A_1; namely, he sends $[RA_{A_2} = w \oplus k_{B_1A_2} \oplus k_{B_3A_2}; A_2]$ to B_1, which again will partially decipher it to get $m_2 = w \oplus k_{B_3A_2}$. The interesting part now, is how B_1 is going to compare the advertisements of A_1 and A_2. To do so, B_1 compares the aggregation secret $as_{A_1A_2}$ with the advertisements sent by A_1 and A_2. More precisely, B_1 first performs an XOR operation between m_1 and m_2. If A_1 and A_2 are advertising the same word w then thanks to XOR's properties, we have:

$$m_1 \oplus m_2 = k_{B_3A_1} \oplus k_{B_3A_2} = as_{A_1A_2}.$$

This simple test allows B_1 to determine whether the advertisements of A_1 and A_2 are equivalent or not, and allow him to aggregate these information and update his forwarding table as follows:

$$(w \oplus k_{B_3A_1} \Leftrightarrow w \oplus k_{B_3A_2}) \rightarrow A_1, A_2.$$

Table I summarizes all the aggregation process at node B_1. This process is always the same at each other node but with different inputs.

After building the forwarding table, advertisements are propagated upwards. Once an encryption layer is removed and some equivalence verification are performed, the intermediate node adds a new encryption layer by using the key it shares with its grandparent. If the advertisements were detected as being equivalent, then the intermediate node only forwards one of them. For example, node B_1 forwards $[m'_1 = m_1 \oplus k_{B_1B_4} = w \oplus k_{B_3A_1} \oplus k_{B_1B_4}; A_1]$ to node B_3. This last information is important so that B_3 knows which key to use to remove part of the encryption, namely $k_{B_3A_1}$, so B_3 computes $m''_1 = m'_1 \oplus k_{B_1B_4} = w \oplus k_{B_1B_4}$.

At this level we can describe the case where a node and its children share the same interest. Suppose that B_1 shares a common interest with its children. Then, there is a small work around concerning aggregation: B_1 does not aggregate his advertisement with its children advertisement. Indeed, if B_1 was able to detect by any way that its own interest is the same as A_1, it would know what A_1 is interested in and hence violate A_1's privacy. By sending the message twice (once for him, and once for all nodes downstream) we protect privacy by adding little complexity over just one link. B_3 is indeed capable of detecting that the two messages sent by B_1 correspond to the same word w. The message that B_1 sends to advertise its interest is $[RA_{B_1} = w \oplus k_{B_1B_3} \oplus k_{B_1B_4}; B_1]$ which B_3 then modifies by removing their shared key giving $m_3 = RA_{B_1} \oplus k_{B_1B_3} = w \oplus k_{B_1B_4}$ which is equal to m''_1, so B_3 immediately detects that the two advertisements correspond to the same word and updates its forwarding table by putting the two advertisements on the same row.

The process of adding and removing encryption layers at each node goes forward until it arrives to C which is the only node which is able to decipher the advertisement w. Table II describes the propagation of A_1's advertisement from A_1 to C.

So far, so good, but what happens if a node has more than two children and want to compare their advertisements. Suppose for example, that A_3, A_4 and A_5 are all interested

Receiving advertisements:
$A_1 \rightarrow B_1$: $[RA_{A_1} = w \oplus k_{B_1A_1} \oplus k_{B_3A_1}; A_1]$
$A_2 \rightarrow B_1$: $[RA_{A_2} = w \oplus k_{B_1A_2} \oplus k_{B_3A_2}; A_2]$

Advertisement processing: remove an encryption layer
$m_1 = RA_{A_1} \oplus k_{B_1A_1} = w \oplus k_{B_3A_1}$
$m_2 = RA_{A_2} \oplus k_{B_1A_2} = w \oplus k_{B_3A_2}$

Equivalence check:
$m_1 \oplus m_2 \stackrel{?}{=} as_{A_1A_2}$

Forwarding table updating:
B_1 adds to its forwarding table FT_1 the following row:
$(w \oplus k_{B_3A_1} \Leftrightarrow w \oplus k_{B_3A_2}) \rightarrow A_1, A_2$

Advertisement forwarding: add an encryption layer
$B_1 \rightarrow B_3$: $m'_1 = m_1 \oplus k_{B_1B_4} = w \oplus k_{B_3A_1} \oplus k_{B_1B_4}$

TABLE I
FORWARDING TABLE BUILDING AND ADVERTISEMENT FORWARDING IN NODE B_1

Step	Advertisement message
A_1	w
$A_1 \rightarrow B_1$	$[w \oplus k_{B_1A_1} \oplus k_{B_3A_1}; A_1]$
B_1	$w \oplus k_{B_3A_1}$
$B_1 \rightarrow B_3$	$[w \oplus k_{B_3A_1} \oplus k_{B_1B_4}; A_1]$
B_3	$w \oplus k_{B_1B_4}$
$B_3 \rightarrow B_4$	$[w \oplus k_{B_1B_4} \oplus k_{B_3C}; B_1]$
B_4	$w \oplus k_{B_3C}$
$B_4 \rightarrow C$	$[w \oplus k_{B_3C} \oplus k_{B_4C}; B_3]$
C	w

TABLE II
PROPAGATION OF AN ADVERTISEMENT FROM A_1 TO C

in a word w'. In this case, the intermediate node B_2 simply does comparisons two by two, with the associated pair-wise aggregation secrets: $as_{A_3A_4}$, $as_{A_3A_5}$ and $as_{A_4A_5}$. Then B_2 detects equivalences between advertisements and propagates finally just one advertisement to B_3, say the one of A_3. Table III presents the forwarding table of B_3 which is interesting since it presents two rows with an aggregation between the advertisements of a child and grandchild (B_1 and A_1). It is worth noting also that the forwarding table keeps not only the next hop, but also the hop after (when there is one) because it influences the way data is encrypted when information returns downstream.

$w \oplus k_{B_1B_4} \rightarrow B_1, B_1(A_1)$ $w' \oplus k_{B_2B_4} \rightarrow B_2(A_3)$

TABLE III
FORWARDING TABLE OF NODE B_3

D. Content distribution and secure look-up

Now that the advertisement propagation process has been detailed, we explain the content distribution algorithm. This algorithm roughly follows the advertisement process in the reverse path. When C receives an advertisement, it can publish matching content and send it downstream. It first chooses randomly a secret key k_C and encrypts the payload P with it (using any symmetric encryption algorithm like AES ([1]) for example, we will denote this symmetric encryption algorithm by $\mathcal{E}$ for the rest of the paper.). It then encrypts the key k_C and the keyword matching the advertisement w in the same way as the receivers did for their advertisements, that is to say by XORing them with the key it shares with its child and grandchild which are on the path of interested receivers. In the example, the message sent by C is as follows:

$$PC = [w \oplus k_{B_3C} \oplus k_{B_4C}; k_C \oplus k'_{B_3C} \oplus k'_{B_4C}; \mathcal{E}_{k_C}(P)].$$

Since the security of one time pads relies on the unique use of the encryption key, k_C is encrypted with k' keys instead of k keys. In the message PC, the first part is the control information CI that allows intermediate nodes to do look-up and hence correctly forward them, the second one allows interested nodes to have access to the encryption key and the last one is the payload. When an intermediate node receives the message, it first uses the first part to do the look-up and take a forwarding decision. B_4 for instance, extracts from CI the value $w \oplus k_{B_3C}$ by eliminating k_{B_4C}. It then looks in his forwarding table if he has an entry that matches it, and finds out that B_3 is interested in this content not for himself but for B_1. It then needs to forward him the message in this optic, by modifying the control message and the key and sending them to B_3 as follows:

$$[w \oplus k_{B_3C} \oplus k_{B_4B_1}; k_C \oplus k'_{B_3C} \oplus k'_{B_4B_1}; \mathcal{E}_{k_C}(P)].$$

The payload is not modified, only the control information and the encrypted key are modified with the multiple encryption system and follow the reverse path of advertisements so that they can easily be processed by intermediate nodes and eventually reach their destination. Table IV shows how the look-up and forwarding operation are performed at node B_3 and table V describes the evolution of a content published by C and forwarded to A_1.

Receiving of content: $B_4 \rightarrow B_3$: $[w \oplus k_{B_3C} \oplus k_{B_4B_1}; k_C \oplus k'_{B_3C} \oplus k'_{B_4B_1}; \mathcal{E}_{k_C}(P)]$
Message processing: $[w \oplus k_{B_4B_1}; k_C \oplus k'_{B_4B_1}; \mathcal{E}_{k_C}(P)]$
Secure look-up with forwarding table FT_3 (table III): $w \oplus k_{B_1B_4} \rightarrow B_1, B_1(A_1)$
Construction and forwarding of messages: $B_3 \rightarrow B_1$: $[w \oplus k_{B_4B_1} \oplus k_{B_3A_1}; k_C \oplus k'_{B_4B_1} \oplus k'_{B_3A_1}; \mathcal{E}_{k_C}(P)]$ $B_3 \rightarrow B_1$: $[w \oplus k_{B_4B_1} \oplus k_{B_3B_1}; k_C \oplus k'_{B_4B_1} \oplus k_{B_3B_1}; \mathcal{E}_{k_C}(P)]$

TABLE IV
SECURE LOOK-UP AND FORWARDING AT NODE B_3

Step	Content message
C	$[w; k_C; P]$
$C \rightarrow B_4$	$[w \oplus k_{B_3C} \oplus k_{B_4C}; k_C \oplus k'_{B_3C} \oplus k'_{B_4C}; \mathcal{E}_{k_C}(P)]$
B_4	$[w \oplus k_{B_3C}; k_C \oplus k'_{B_3C}; \mathcal{E}_{k_C}(P)]$
$B_4 \rightarrow B_3$	$[w \oplus k_{B_3C} \oplus k_{B_4B_1}; k_C \oplus k'_{B_3C} \oplus k'_{B_4B_1}; \mathcal{E}_{k_C}(P)]$
B_3	$[w \oplus k_{B_4B_1}; k_C \oplus k'_{B_4B_1}; \mathcal{E}_{k_C}(P)]$
$B_3 \rightarrow B_1$	$[w \oplus k_{B_4B_1} \oplus k_{B_3A_1}; k_C \oplus k'_{B_4B_1} \oplus k'_{B_3A_1}; \mathcal{E}_{k_C}(P)]$
B_1	$[w \oplus k_{B_3A_1}; k_C \oplus k'_{B_3A_1}; \mathcal{E}_{k_C}(P)]$
$B_1 \rightarrow A_1$	$[w \oplus k_{B_3A_1} \oplus k_{B_1A_1}; k_C \oplus k'_{B_3A_1} \oplus k'_{B_1A_1}; \mathcal{E}_{k_C}(P)]$
A_1	$[w; k_C; P]$

TABLE V
EVOLUTION OF A MESSAGE PUBLISHED BY C ON ITS PATH TO A_1

E. Evaluation

In this section, we evaluate the security and the performance of the scheme.

We first show that the proposed encryption mechanism with multiple encryption layers ensures confidentiality against external attackers that do not participate to any networking or security operation and further show that it is reaching privacy at level 3.

In a work evaluating the security of cryptosystems in the multi-user setting [2], Bellare et al. have essentially shown that if a cryptosystem is secure in the sense of indistinguishability, then the cryptosystem in the multi-user setting, where related messages are encrypted using different keys, is also secure. This result can be applied to the proposed scheme using the XOR operation as an encryption. When a message is encrypted with two keys it is at least as secure as any individual encryption. Thus, the scheme is at least as secure as a one layer encryption.

Moreover, the security of encryption operation that simply is a XOR depends on the unique utilization of the encryption key. Since the encryption key at each node is only used once and is updated for each new message (either advertisement or content), the operation is perfectly secure. Yet one has to take care of not reusing the same keys for other advertisement. This problem can easily be solved by updating the keys in a decentralized way : each node can just compute a hash of the keys it owns. For example $k_{A_1B_1}$ would become $h(k_{A_1B_1})$ and this is enough to maintain the security of the scheme.

We now show that the proposed framework ensures the third level privacy model whereby every node becomes a potential adversary and thus intermediate nodes are not trusted. Indeed, thanks to the use of multiple encryption layers, the confidentiality of messages relies on the use of keys belonging to different users. Messages are namely forwarded and continuously modified by the addition and removal of encryption layers but they remain unaccessible to intermediate nodes at all times, even if these nodes have the same interest. In the proposed framework, the security mechanism presented relies on the use of two encryption layers in order to simplify its description. However it also means that if two consecutive nodes, a node and its parent, collude and hence share their own keying material, they can decrypt their children nodes' interest. In order to prevent such attacks, the number of encryption layers can be increased as described in [7]. Therefore, the privacy of the scheme depends on the choice of the number of encryption layers denoted by m. The larger values for m imply a larger number of nodes to collude to break it. However, if m is very large, then the number of keys stored at each node becomes very large and the key distribution protocol can have an impact on the performance of the protocol. The choice of m is hence a trade-off that depends on the scenario and the topology of the network.

We now evaluate the performance of the scheme in terms of memory storage and computational cost.

First of all, the computational activity of each node for both the encryption and decryption operations is only a simple XOR. The memory cost is related to the key distribution algorithm: each node shares two keys with its parent and grandparent and two keys with each of its children and grandchildren. It also stores a small number of aggregation secrets that depends on the number of their children.

Furthermore, thanks to the secure aggregation of advertisements, forwarding tables are also optimized. Indeed, any intermediate node is able to compare encrypted advertisements and discover equivalences in order to optimize its forwarding table, which also improves the performance of the look-up operation.

To put it in a nutshell, this scheme enforces, at a very low cost, full privacy all the way since intermediate nodes (and even the root C) do not know what is the final destination of the information (except the node before it), they just know in which direction to forward the packet. The only point of weakness of this scheme is that it requires a very particular configuration.

VI. Conclusion & Future work

In this paper we presented the analysis of privacy issues in content-based networking. We defined three privacy models that adapt to different networking scenarios and achieve different levels of privacy. We also identified two main security primitives which are necessary to secure content-based networking operations, namely **secure look-up** and **building of forwarding tables**, and we have detailed the requirements that each of these primitives should fulfill in order to fit in each privacy model.

Finally, we presented an original approach based on multiple encryption that achieves full privacy content-based networking. This scheme preserves privacy of receivers very efficiently and has a very low cost since all encryption operations are simple XORs.

As a future work, we intend to develop this scheme by improving its flexibility regarding the network topology and the advertisements format. We would like indeed to extend receiver advertisements to the form of disjunction of conjunction of several interests.

References

[1] Advanced encryption standard. Federal Information Processing Standards Publication 197, November 2001.

[2] M. Bellare, A. Boldyreva, and S. Micali. Public-key encryption in a multiuser setting: Security proofs and improvements. In *Eurocrypt 2000*, volume LNCS 1807, pages 259–274. Springer Verlag, 2000.

[3] A. Carzaniga, M. J. Rutherford, and A. L. Wolf. A routing scheme for content-based networking. In *IEEE INFOCOM 2004*, Hong Kong, China, March 2004.

[4] A. Carzaniga and A. L. Wolf. Forwarding in a content-based network. In *SIGCOMM*, pages 163–174, 2003.

[5] L. Lilien, Z. Kamal, V. Bhuse, and A. Gupta. Opportunistic networks: The concept and research challenges in privacy and security. In *NSF Intl. Workshop on Research Challenges in Security and Privacy for Mobile and Wireless Networks (WSPWN 2006)*, Miami, March 2006.

[6] H. A. Nguyen, S. Giordano, and A. Puiatti. Probabilistic routing protocol for intermittently connected mobile ad hoc network (propicman). *World of Wireless, Mobile and Multimedia Networks, 2007. WoWMoM 2007. IEEE International Symposium on a*, pages 1–6, 18-21 June 2007.

[7] M. Önen and R. Molva. Secure data aggregation with multiple encryption. In *Wireless Sensor Networks, 4th European Conference, EWSN 2007*, Lecture Notes in Computer Science, pages 117–132, Delft, The Netherlands, 29-31 January 2007. Springer.