A-Kad: an anonymous P2P protocol based on Kad network

YongQing Ni, DaeHun Nyang
Information Security Research Lab
Inha University
Incheon 402-751, Korea
niyongqing@gmail.com, nyang@inha.ac.kr

Xu Wang
Information Security Research Lab
Inha University
Incheon 402-751, Korea
wmingxuan@hotmail.com

***Abstract*—With the growth of decentralized network users, preserving privacy becomes a critical issue in this open community. Kad-based network, as a typical decentralized system, has been widely used nowadays. However, there is not enough research to achieve anonymity on it. In this paper, we propose an anonymous protocol based on Kad network, named Anonymous Kad (A-Kad), which achieves complete privacy and security for file providers and requesters. A-Kad has the desired property of anonymity and still keeps high efficiency in publishing and querying phases. To achieve anonymity, we establish two anonymous channels which help file providers to anonymously publish file information and securely transfer files. Through these two channels, the file requester can also efficiently query and retrieve files without worrying about exposing its behavior. Moreover, we propose an anonymity degree evaluation model (ADEM) according to three different attacking capabilities and anonymity degree.**

***Index Terms*—anonymity; P2P; Kad; privacy;**

I. Introduction

Nowadays, various internet services have arisen and the interactions between different users have become more frequent than ever. P2P network, a creative model with distributed routing architecture, has become a popular file sharing system recently. More users are willing to join this open community to share and retrieve information or files. Therefore, preserving-privacy causes a wide concern and becomes a critical issue. Anonymity, a major method for protecting privacy, also has proven to be an effective way to prevent personal information leak in P2P network. Thus, we propose a new approach to achieve anonymity of P2P users.

Kademlia, a significant distributed hash table proposed by P. Maymounkov and D. Mazieres [1], has been applied to the Emule system, named Kad network, and is widely used recently. As a highly efficient routing protocol with self-organizing, scalable and robust properties, it is getting more and more attention in P2P network. But to our best knowledge, there is a little contribution in providing anonymity in Kad although questions arise concerning its security. M. Steiner *et al.* presented a global view of Kad [2] with their own crawler which they explored for six months. Besides, in [3], they not only showed misuses and DDos attack on Kad, but also proposed a solution to prevent Sybil attack. W. Peng *et al.* designed their own model of attack on Kad, after comparing the existing attack schemes, such as Sybil and index poisoning attack. They argued that their scheme was more efficient and effective. I. Baumgart and S. Mies gave an example [4] which is a secure key-based routing protocol based on Kademlia [1]. The evaluation of their scheme showed more resilience than the original Kademlia [1] but is still vulnerable to being mislead by an adversary. Because our main goal of this paper is to achieve anonymity, we won't go into more detail here.

In this paper, we propose an anonymous P2P protocol based on Kad network (A-Kad) by establishing two anonymous channels for the publishing and file retrieving phases. According to existing research, many researchers tended to achieve mutual anonymity both for file providers and requesters. SSMP [5], a mutual anonymity protocol based on Shamir's secret sharing scheme, provides privacy protection for the requester by distributing different pieces of a secret share to different nodes and only the nodes who collect enough number of partial secret shares can recover the plain query. V. Scarlata et.al also proposed a mutual anonymous P2P file sharing protocol named APFS [6]. They provided two variants based on their basic model, one is a unicast communication channel with a central coordinator to bootstrap, while the other avoids using the central coordinator point by utilizing multicast routing. Comparing these two different versions, the latter has a larger advantage of strong anonymity than the unicast one. Additionally, A. Singh and L. Liu introduced a new anonymous service over a structured P2P network, named Aayaat [7], by adding clouds topology on top of DHT-based P2P network. They trigger an anonymous query in clouds before sending the message which can help hide identities of requesters. Among all above schemes that we mentioned, most of them succeeded in attaining mutual anonymity between initiator and responder. According to these models, all security analysis is based on a local attacker and without considering a possible global attacker who is omnipresent and has full access to an entire network. In this paper, we propose the A-Kad protocol with strong anonymity and investigate its capabilities according to ADEM model.

Our paper is organized as follows. We firstly introduce background of Kademlia [1] and Onion Routing [8] in Section II. In Section III, we describe our design of A-Kad protocol in detail. We carry out analysis of anonymity and performance in

978-1-4244-5113-5/09/$25.00 ©2009 IEEE

Section IV and V. Finally, we make a conclusion and indicate our future work.

II. Background

A. Kademlia

Compared to Chord [9], Kademlia [1] has several advantages, such as a novel XOR metric for node distance calculation, can be widely applied to Emule system and is insusceptible to several known common attacks. For these reasons, Kademlia [1] arouses wide concern nowadays.

Same as Chord [9], Kademlia [1] also assigns a unique NodeID in 128-bit to each node as its identity in the whole P2P network. However, different from Chord [9], Kademlia [1] takes advantage of XOR metric to calculate the distance between different nodes based on an individual NodeID. Each node maintains a routing table consisting of up to 128 k-buckets. Every bucket contains at most k contacts with ⟨IP, UDP port, NodeID⟩. Extended to a publish or query scheme, each file will also have a unique FileID which has the same length as the NodeID. The file information will be published to the nodes who have the same or similar NodeID to FileID. In addition, to enhance the search efficiency, each node has several corresponding keywords and each keyword also has a unique hash value which constructs a key-value pair. This brilliant design provides a highly efficient publish and search scheme.

B. Onion Routing

Onion Routing [8] is a general protocol which provides anonymous communication over public network on condition that requester knows the public keys of all the other nodes. To set up the anonymous channel, the requester selects a random routing path through CORs [8] and creates an onion layer by layer with corresponding public keys. By doing this, the message can be unwrapped by equivalent COR who processes the corresponding private key and forwards it to the node that belongs to the inner layer. Onion Routing, however, is originally vulnerable to a single malicious node recoding traffic and compromising successive nodes in the routing circuit. R. Dingledine *et al.* designed Tor [10], the second generation of Onion Routing. Instead of using a single multiply encrypted data structure to lay each circuit, Tor [10] uses an incremental path to build each successive hop in the circuit.

III. A-Kad protocol design

In this section, we describe the design of A-Kad. The purpose of our protocol is to provide an anonymous protocol based on Kad network. With such a system, we argue, the identities of users will be well protected no matter if they are file providers or requesters. To achieve the anonymity, we consider four aspects of our main goal for the A-Kad. *(1) The file provider can anonymously publish its file information without worrying that its identity will be revealed. (2) The file requester can send its query anonymously without worrying that its querying content will be exposed. (3) The file provider can securely transfer file and its privacy will be well protected. (4) The file requester also can safely retrieve the files that it wants and no one can know the specific file contents.*

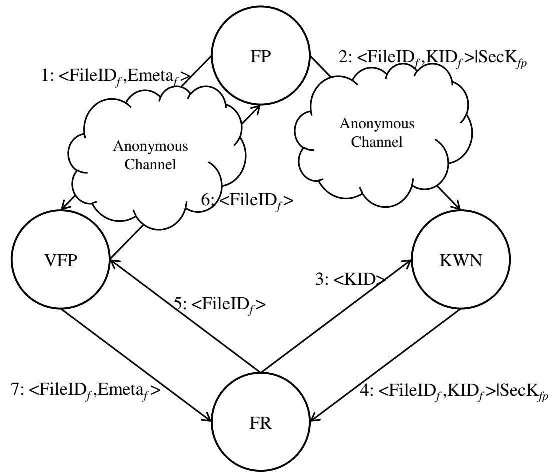

Fig. 1. A-Kad model

To achieve the above goal, we propose three approaches based on the original Kad: *(1) Establish two anonymous channels for file providers. (2) Encrypt the Meta information. (3) Replace the key word query with a hash value.* Figure 1 shows the whole architecture of A-Kad protocol. We describe it more detail in the coming sections.

The following notations and entities are used:
NodeID stands for the identity of a node which is a 128-bit hash value generated when each node joins the P2P network. FileID is also a hash value of a file with the same length of NodeID. KID denotes the hash value of a key word which also has a 128-bit length. We use FP and FR to denote the file provider and requester, respectively. VFP denotes virtual file provider whose NodeID is the same or similar to the FileID. KWN means key word node whose NodeID is the same or similar to the KID. We use Meta to denote file information such as file name, type, size etc. A session key between node i and j is denoted by SK$_{i,j}$. Encryption of a message M by a session key is given by SK(M). SecK stands for secret key which is used to encrypt the Meta as a symmetric key. In addition, we use Tag as a vector ⟨NodeID, Timestamp⟩ to record interacting history between nodes. X $\stackrel{ac}{\to}$ Y : M represents X sending a message M to Y through the anonymous channel ac. We use g to present a generator of a multiplicative group of prime and g^x denotes a random exponent.

A. Publishing Phase

In original Kad network, Meta is directly published to VFP. By doing so, VFP knows all the public information of the FP, such as the IP, file name etc, which means there is no privacy protection for the file provider.

To achieve the complete anonymity for the FP, we consider three terms of privacy: *1. The successor of FP cannot know what kind of file is FP providing; 2. The FR cannot identify* *who provides the file; 3. A global attacker cannot obtain the file information and detect the source provider.*

1) Anonymous channel establishment:

Step 1: FP randomly selects several CORs [8] as an candidate onion router when it is updating its bucket. Here we assume two hops are chosen by FP such as node A and B.

Step 2: With Diffie-Hellman(DH),FP establishes session keys with A:

$$\begin{aligned} \text{FP} \rightarrow \text{A} &\quad : \quad g^{fp1} \\ \text{A} \rightarrow \text{FP} &\quad : \quad g^{a} \end{aligned}$$

By doing so, FP can construct session key with A; $\text{SK}_{fp,a}$ = $g^{fp1 \cdot a}$. With this session key, FP exchanges DH handshake with B:

$$\begin{aligned} \text{FP} \rightarrow \text{A} &\quad : \quad \text{SK}_{fp,a}(g^{fp2}, B) \\ \text{A} \rightarrow \text{B} &\quad : \quad g^{fp2} \\ \text{B} \rightarrow \text{A} &\quad : \quad g^{b} \\ \text{A} \rightarrow \text{FP} &\quad : \quad \text{SK}_{fp,a}(g^{b}) \end{aligned}$$

FP decrypts the message from A and constructs session key with B; $\text{SK}_{fp,b}$ = $g^{fp2 \cdot b}$. Figure 2(1-6) shows the procedure of this step.

Step 3: With the established session keys, FP sends the half of DH hadshake g^{fp3} through selected CORs, A and B:

$$\begin{aligned} \text{FP} \rightarrow \text{A} &\quad : \quad \text{SK}_{fp,a}(\text{SK}_{fp,b}(g^{fp3}, \text{VFP}), \text{B}) \\ \text{A} \rightarrow \text{B} &\quad : \quad \text{SK}_{fp,b}(g^{fp3}, \text{VFP}) \\ \text{B} \rightarrow \text{VFP} &\quad : \quad g^{fp3} \end{aligned}$$

It is important to note that each COR will record its predecessor's information in $\langle$NodeID,Timestamp$\rangle$ as Tag information when there is an interaction. Finally, g^{fp3} will reach to VFP. By doing this, VFP can construct the secure session key: $\text{SK}_{fp,vfp}$ = $g^{fp3 \cdot vfp}$, although it cannot reveal the specific identity of FP.

Step 4: VFP replies g^{vfp} to B. According to B's Tag information, B encrypts the message with $\text{SK}_{fp,b}$ and forwards it to A. Here, we wrap the onion layer by layer to encrypt the message, naming it wrapping onion router (WOR). After the message reaches the FP, it can easily decrypt it with its session keys and obtains $\text{SK}_{fp,vfp}$ = $g^{fp3 \cdot vfp}$:

$$\begin{aligned} \text{VFP} \rightarrow \text{B} &\quad : \quad g^{vfp} \\ \text{B} \rightarrow \text{A} &\quad : \quad \text{SK}_{fp,b}(g^{vfp}, \text{VFP}) \\ \text{A} \rightarrow \text{FP} &\quad : \quad \text{SK}_{fp,a}(\text{SK}_{fp,b}(g^{vfp}, \text{VFP}), \text{B}) \end{aligned}$$

Thus, it is not just simply used by onion routers. We adopt a reversal way to construct an onion and successfully establish an anonymous channel *ac* between FP and VFP without exposing any identity.

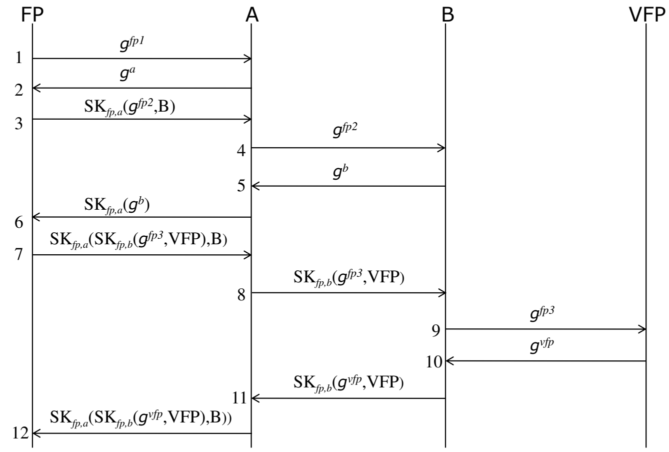

Fig. 2. Anonymous channel establishment

2) File information publishing phase:

Step 1: With the anonymous channel, FP can publish its file information securely and anonymously. FP encrypts Meta_f with SecK_{fp}, gets EMeta_f. After that, FP constructs a vector $\langle\text{FileID}_f$, $\text{EMeta}_f\rangle$. Then, this message will be sent to VFP through the established channel:

$$\begin{aligned} \text{FP} \rightarrow \text{A} &\quad : \quad \text{SK}_{fp,a}(\text{SK}_{fp,b}(\text{SK}_{fp,vfp}(\langle\text{FileID}_f, \text{EMeta}_f\rangle), \text{VFP}), \text{B}) \\ \text{A} \rightarrow \text{B} &\quad : \quad \text{SK}_{fp,b}(\text{SK}_{fp,vfp}(\langle\text{FileID}_f, \text{EMeta}_f\rangle), \text{VFP}) \\ \text{B} \rightarrow \text{VFP} &\quad : \quad \text{SK}_{fp,vfp}(\langle\text{FileID}_f, \text{EMeta}_f\rangle) \end{aligned}$$

After VFP receives the message forwarded by B, it decrypts the message and stores the vector in its bucket.

Based on the original Kad, FP publishes the source NodeID and $\langle$KID, Keyword$\rangle$ as a key-value pair to KWN. By doing this way, it enhances the search efficiency when the users launch keyword queries. We should notice that this scheme helps FP publish its file references efficiently but without any privacy protection. Thus, we use KID as query content instead of keyword plaintext.

Step 2: By applying the same publishing scheme as *Step 1*, FP issues $\langle\text{FileID}_f,\text{KID}_f\rangle|\text{SecK}_{fp}$ through the anonymous channel

$$\text{FP} \xrightarrow{ac} \text{KWN} \quad : \quad \text{SK}_{fp,kwn}(\langle\text{FileID}_f, \text{KID}_f\rangle \mid \text{SecK}_{fp})$$

KWN decrypts the message with session key $\text{SK}_{fp,kwn}$ and stores the vector $\langle\text{FileID}_f,\text{KID}_f\rangle$ and SecK_{fp} in its bucket.

B. Query Phase

In the previous phase, all the file information has been published to the P2P network on condition that the privacy of FP is well protected. In this section, we describe how a user can launch a query anonymously and obtain the response efficiently.

Step 1: Before sending a query, FR hashes its keyword as query content and sends $\langle$KID$\rangle$ to KWN:

$$\text{FR} \rightarrow \text{KWN} \quad : \quad \langle \text{KID} \rangle$$

Step 2: KWN will check and route its bucket. If there's a match with KID, it will reply as follows(Here, we assume the secret key will be delivered in SSL layer). This step is shown in Fig. 1(4):

$$\text{KWN} \rightarrow \text{FR} \quad : \quad \langle \text{FileID}_f, \text{KID}_f \rangle \mid \text{SecK}_{fp}$$

Step 3: FR sends the query FileID_f to VFP, if there is a match in VFP's bucket, FR is able to receive $\langle \text{FileID}_f$,$\text{EMeta}_f \rangle$ from VFP. Therefore, it decrypts the EMeta_f with SecK_{fp} and decides whether it will download or not. If yes, FR replies file requesting message to VFP.

C. File transferring phase

After VFP receives a confirming message from FR, it will start to trigger the file transfer:

Step 1: VFP decrypts FR's confirming message with $\text{SK}_{fp,vfp}$ and forwards it to B based on its Tag information.

$$\begin{aligned} \text{VFP} \rightarrow \text{B} \quad &: \quad \text{SK}_{fp,vfp}(\langle \text{FileID}_f, \text{Req}_f \rangle) \\ \text{B} \rightarrow \text{A} \quad &: \quad \text{SK}_{fp,b}(\text{SK}_{fp,vfp}(\langle \text{FileID}_f, \text{Req}_f \rangle)) \\ \text{A} \rightarrow \text{FP} \quad &: \quad \text{SK}_{fp,a}(\text{SK}_{fp,b}(\text{SK}_{fp,vfp}(\langle \text{FileID}_f, \text{Req}_f \rangle))) \end{aligned}$$

Step 3: When FP receives the requesting message, it encrypts the file package with SecK_{fp}. Then, through the established anonymous channel, FP transfers the encrypted data EData_f to A according to its Tag information. Through the anonymous channel *ac*, the EData_f will reach VFP.

$$\text{FP} \overset{ac}{\rightarrow} \text{VFP} \quad : \quad \langle \text{EData}_f \rangle$$

Step 4: VFP checks its Tag information and directly forwards the data to FR:

$$\text{VFP} \rightarrow \text{FR} \quad : \quad \langle \text{EData}_f \rangle$$

Step 5: After receiving the encrypted package, FR decrypts it with SecK_{fp} and obtains the plaintext.

Note that we recommend to use multiple anonymous channels to enhance the network reliability although we described it only using a single channel.

IV. Anonymity analysis

In this section, we focus on possible privacy compromising attacks on A-Kad. We start to perform our analysis on four existing channels established in A-Kad protocol.

A. Channel between FP and VFP

As we mentioned in Section III, FP publishes Meta and transfers files through this channel. By establishing session keys with CORs and VFP, FP can anonymously publish Meta without worrying about its identity being detected. In this phase, we consider three types of possible attacks. First, the man-in-the-middle attack. Since the channel between FP and VFP is completely encapsulated with session keys, the man-in-the-middle attack cannot monitor the traffic between FP and VFP. Second, the global attacker, the same scheme as we mentioned above, it can only detect the incoming and outgoing traffic but cannot know where the destination of the traffic is. Third, we assume VFP is compromised. In this case, although VFP knows the $\langle \text{FileID}_f$,$\text{EMeta}_f \rangle$ it cannot reveal who is the source provider. In terms of file transferring phase, FP encrypts the file package with its own symmetric key, which means no one can recover the plaintext except those who have the decryption key.

B. Channel between FP and KWN

Through this channel, KWN indirectly receives $\langle \text{FileID}_f$,$\text{KID}_f \rangle$|SecK_{fp} from FP. We notice that KWN only obtains a file reference and the corresponding symmetric key in the whole interaction. There is no any leakage of information leak regarding the file provider itself.

C. Channel between FR and KWN

Note that our main goal of designing this channel is to achieve and preserve the privacy of FR. Instead of using a keyword as searching content, we use a unique hash value KID which corresponds to a specific keyword as a query. By doing this, the man-in-the-middle attack is prevented even though the adversary can eavesdrop the specific query. Besides, KWN cannot reveal the specific keyword because it only knows the KID.

D. Channel between FR and VFP

This channel plays two important roles for FR, one is for issuing a FileID query and file downloading request, the other is to transfer files. Even though the attacker observes on this channel when FR issues a query, according to the FileID, the eavesdropper cannot detect which file is queried or requested by FR.

After analyzing the above four channels, we notice that a completely anonymous P2P protocol must satisfy the following three properties [11]:

- **Unlinkability:** It means no message is linkable to a particular sender-receiver pair. Moreover, the adversary is unable to relate the node's identity, message and behavior.
- **Unidentifiability:** The adversary is unable to discern a node's identity, behavior or other related information when the node acts as a file provider or requester. This property consists of sender anonymity and receiver anonymity. Sender anonymity means a specific message

cannot be linked to a corresponding sender identity. Similarly, receiver anonymity prevents the specific message from being linked to a specific receiver identity.

- **Unobservability:** The adversary is unable to observe items of interest which includes most of the information, such as the nodes' identities, messages, related information, and traffic. This capability implies anonymity by keeping message indistinguishable from different entities. It also can be defined into sender and receiver *Unobservability*.

Meanwhile, the adversary is omnipresent and its capabilities range from weak to strong. To our best knowledge, we list three typical capabilities possessed by possible adversaries:

- **Reachability:** With *Reachability*, a global adversary is omnipresent and able to access to the whole network. A local adversary can succeed in accessing a part of the network. This means the adversaries can learn complete or limited information about the network they are interested in.
- **Attackability:** This capability corresponds to an adversary identifying the sender or receiver with a specific message by tracing a message link or disrupting the whole system. An passive attacker with this capability can only eavesdrop messages by observing traffic while an active attacker may modify or forge messages.
- **Adaptability:** *Adaptability* means an adversary is dynamic and able to collect information and behaviors from compromised nodes. With this capability, an attacker can utilize all available information to infer who is the sender or receiver. Moreover, the attacker can also use a prescheduled plan to launch malicious operation or attack.

To better describe the relations between anonymity capabilities and the threaten model, we present anonymity degree as a standard anonymity evaluation criteria. We will carry out more detailed analysis based on our A-Kad protocol.

In our protocol, there are four channels established and two of them have the *Unlinkability*: the channel FP-VFP and FP-KWN. For a global adversary, it can only detect certain links between the processor and successor but cannot connect the whole link of the two channels: FP-VFP and FP-KWN. On the other side, the remaining two channels are reachable even for the local adversary. Therefore, we achieve the *Unlinkability* of FP but not FR.

In terms of *Unidentifiability*, we should consider both FP and FR in A-Kad protocol. On the view of a global adversary, it can learn the identity of FR but cannot detect who FP is since two of the anonymous channels can prevent FP from the *Reachability* of the adversary. Thus, with *Unlinkability*, FP cannot be identified by the adversary even though it has *Attackability* to trace certain messages. Meanwhile, if the adversary can dynamically trace and collect information from enough compromised nodes, it can identify FP. However, this case may potentially happen only if the adversary can infect a large scale of nodes.

TABLE I
ANONYMITY DEGREE COMPARISON

Anonymity Degree \ Protocol		Freenet		Salsa		A-Kad	
		FP	FR	FP	FR	FP	FR
Unlink-ability	Reachability	Yes	No	Yes	Yes	Yes	No
	Attackablity	Yes	No	Yes	Yes	Yes	No
	Adaptability	Yes	No	Yes	Yes	Yes	No
Unidentifi-ability	Reachability	Yes	Yes	No	Yes	Yes	Yes
	Attackablity	No	Yes	No	No	Yes	Yes
	Adaptability	No	No	No	No	Yes	Yes
Unobserv-ability	Reachability	No	No	No	Yes	Yes	Yes
	Attackablity	No	No	No	No	No	No
	Adaptability	No	No	No	No	No	No

Regarding *Unobservability*, a model with this capability can be well protected from the adversary who has *Reachability* and *Attackability*. In A-Kad protocol, we have achieved *Unobservability* for FP by establishing two anonymous channels. Take into account FR, we should notice that even though there is no *Unobservability*, an adversary can hardly observe FR's identity and related information because the only message it can eavesdrop is a hash value.

To conclude our analysis on A-Kad, we choose two typical anonymous P2P protocols, Freenet [12] and Salsa [13] for comparing with our protocol.

Freenet [12] is both an original design for anonymity and an implemented system but it cannot hide the provider of a particular file because a global attacker can easily find all copies. Thus, if an attacker possesses *Attackability*, the identity of FP will be revealed. Besides, there is no anonymous channel established in Freenet [12] which means it does not have the capability of *Unobservability*. In terms of FR, the request is forwarded based on hops-to-live, even though an attacker with *Reachability* can obtain the link to the FR, therefore, there's no *Unlinkability* achieved. In Salsa [13], an initiator(FR) selects a set of nodes and builds a circuit to achieve its anonymity. To apply its capability to our ADEM model, its FR has *Unlinkability* even if an attacker possesses *Attackability*. However, if we check its *Unidentifiability* and *Unobservability*, we find that it can only resist an attacker with *Reachability*. While referring to FP, they only achieved *Unlinkability* for a privacy-preserving concern.

The following table gives a general idea of the anonymity degree provided by the methods analyzed in the above paragraph. It shows that our protocol achieves maximum anonymity degree of FP compared with Freenet and Salsa. However, in order to maintain the high efficiency of query, A-Kad does not perform *Unlinkability* of FR since its privacy is well protected.

V. PERFORMANCE ANALYSIS

We estimate the additional overhead incurred in our A-Kad in this section. A-Kad's overhead is mainly due to establish anonymous channel and is incurred when communication happened in it. In A-Kad protocol, we use Diffie-Hellman key exchange to establish anonymous channels for FP and messages are delivered by wrapping onion routers.

As we mentioned in Section IV, the two anonymous channels are mainly used in *File information publishing phase* and *File transferring phase*, respectively. In former phase, the FP is able to publish its file information to corresponding VFP and KWN simultaneously. By doing so, the latency of publishing is reduced. Meanwhile, the Diffie-Hellman key exchange only happened one time in anonymous channel establishment. The overhead incurred in this phase is reasonable because it appears to be no more noticeable than other delays in practice. In *File transferring phase*, only symmetric encryption was used in A-Kad, the extra overhead is not a big burden if we consider it in the practical network.

VI. Conclusion and Future work

In this paper, we have presented the design of A-Kad, a P2P protocol that provides anonymity based on Kad network. By wrapping Onion Routing, we provided the desired properties of anonymity with two established anonymous channels for the file provider. Additionally, we also maintain high efficiency and privacy-preserving in the searching phase for file requester using hash value query instead of a keyword. Moreover, we proposed an approach to evaluate anonymity degree: anonymity degree evaluation model (ADEM) according to the three different attacking capabilities. Based on the ADEM model, we extensively analyzed the anonymity properties of A-Kad.

As future work, we plan to enhance the anonymity capability of A-Kad on condition of maintaining high efficiency in the querying phase. Moreover, we intend to deliberate certain known attacks like DDos, Sybil attack and intersection attack to enhance security of our protocol in the future implementation. An interesting future research direction is defining performance metrics and the more specific anonymity level evaluation criteria based on our ADEM model.

References

[1] P. Maymounkov and D. Mazieres, “Kademlia: A Peer-to-Peer Information System Based on the XOR Metric,” *Peer-To-Peer Systems: First International Workshop, IPTPS 2002, Cambridge, MA, USA, March 7-8, 2002: Revised Papers*, 2002.

[2] M. Steiner, T. En-Najjary, and E. W. Biersack, “A global view of kad.” in *Internet Measurement Comference*, C. Dovrolis and M. Roughan, Eds. ACM, pp. 117–122.

[3] ——, “Exploiting kad: possible uses and misuses.” *Computer Communication Review*, vol. 37, no. 5, pp. 65–70, 2007.

[4] I. Baumgart and S. Mies, “S/kademlia: A practicable approach towards secure key-based routing.” in *ICPADS*. IEEE Computer Society, 2007, pp. 1–8.

[5] J. Han, Y. Liu, L. Xiao, R. Xiao, and L. M. Ni, “A mutual anonymous peer-to-peer protocol design.” in *IPDPS*. IEEE Computer Society.

[6] V. Scarlata, B. N. Levine, and C. Shields, “Responder anonymity and anonymous peer-to-peer file sharing.” in *ICNP*. IEEE Computer Society, pp. 272–280.

[7] A. Singh, B. Gedik, and L. Liu, “Agyaat: mutual anonymity over structured P2P networks,” *Internet Research*, vol. 16, no. 2, pp. 189–212, 2006.

[8] P. F. Syverson, D. M. Goldschlag, and M. G. Reed, “Anonymous connections and onion routing.” in *IEEE Symposium on Security and Privacy*. IEEE Computer Society, 1997, pp. 44–54.

[9] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan, “Chord: A scalable peer-to-peer lookup service for internet applications,” in *SIGCOMM ’01*. New York, NY, USA: ACM, 2001, pp. 149–160.

[10] R. Dingledine, N. Mathewson, and P. F. Syverson, “Tor: The second-generation onion router.” in *USENIX Security Symposium*. USENIX, 2004, pp. 303–320.

[11] D. Kelly, “A taxonomy for and analysis of anonymous communications networks,” Ph.D. dissertation, Air Force Institute of Technology, March 2009.

[12] I. Clarke, O. Sandberg, B. Wiley, and T. W. Hong, “Freenet: A distributed anonymous information storage and retrieval system,” *Lecture Notes in Computer Science*, pp. 46+.

[13] A. Nambiar and M. Wright, “Salsa: a structured approach to large-scale anonymity.” in *ACM Conference on Computer and Communications Security*, A. Juels, R. N. Wright, and S. D. C. di Vimercati, Eds. ACM, 2006, pp. 17–26.