ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 2, Issue 4, April 2013

A Modified Algorithm to Verify Integrity and Authentication of Data in Virtual Networks

Sandip Kankal[1], Prof. V. P. Kshirsagar[2]

M.E. Student (CSE)[1], HOD (Department of CSE)[2]

Government College of Engineering, Aurangabad, Maharashtra, India.

Absract : Network Virtualization is a technology that enables the creation of logically isolated network partitions over shared physical network infrastructures so that multiple heterogeneous virtual networks can simultaneously coexist over a shared infrastructure. Network Virtualization is considered as a challenging approach to overcome the current ossification of the Internet. A major issue in the Network Virtualization is an efficient virtual network embedding deals with the efficient mapping of virtual nodes and virtual edges on substrate network resources. In this paper we propose the algorithm that verifies the integrity and authenticity of the packet transmitted from source to destination. The algorithm detects the loss of data if any while transmission and also identifies damage to the packet.

Keywords : Network Virtualization, Virtual Network, VN embedding, MAC,HMAC etc.

I. INTRODUCTION

In the modern world Internet is gaining success in designing the way we access and exchange the information. The current Internet architecture supports a number of distributed applications and different types of topologies. However, its popularity has become biggest reason for its further growth. Adopting new architecture or modification of current architecture requires the general agreement among competing stakeholders because of multi-provider nature. Thus, modifications to the Internet architecture have become limited to simple updates only instead of major alterations to the architecture. This is known as ossification of the Internet.

Network Virtualization can be seen as solution to Internet ossification. Network Virtualization allow the combination of computer network resources into a single platform appearing as a single network. Network Virtualization starts with an architecture that decouples the physical from the virtual aspect of network i.e. physical network performs its intended work of forwarding packets, utilizing trusted and well understood routing protocol, however virtual network maintains operational policies, including access control lists, configuration policies and network services. The roles of traditional Internet service providers is decoupled into two independent entities by using Network Virtualization [1] [3] : Infrastructure providers (InPs) perform the work of managing physical network infrastructures whereas Service providers (SPs) are involved in the creation of virtual networks by collecting resources from different multiple infrastructure providers and offer end-to-end services to end users. This investment of expenditure in physical infrastructure will be reduced.

The Network Virtualization concept is introduced as a solution to diversify the Future Internet architecture into separate Virtual Networks (VNs). A virtual network is a collection of virtual nodes and virtual links. The same physical network is shared by the multiple virtual networks. The VN consist of a set of virtual nodes and links that can be assigned to set of specific substrate node and substrate paths respectively. These virtual networks can be assigned to physical infrastructure in an optimal way. In order to operate properly each virtual network supports physical hardware. The Virtual network embedding problem deals with the mapping of virtual network (VN) request, with constraint on virtual nodes and links, onto specific physical nodes and links in the substrate network that has finite resources [1]. A VN request is a set of virtual nodes (with or without CPU requirements) and set of virtual link (with or without bandwidth requirements). As VN request arrives the virtual

ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 2, Issue 4, April 2013

nodes must be mapped to substrate nodes with sufficient CPU requirements and virtual links are mapped to substrate paths satisfying the bandwidth requirement. To increase substrate network resource utilization and InP revenue, efficient and effective assignment of the online VN request is necessary.

During VN embedding the substrate node is mapped to the virtual node on the basis of higher bandwidth and CPU capacity. The VN embedding involves searching the paths for the source or destination pairs of nodes. The transmission of data is done from the source node to the destination node in the network.

The Figure below shows two VN requests that can be mapped onto substrate network [1].

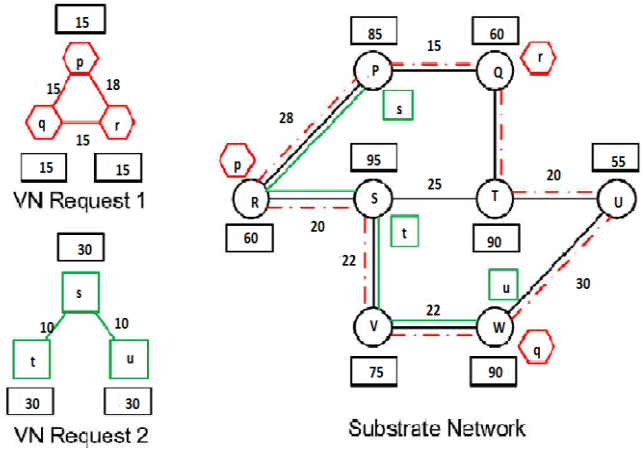

Fig. 1 Mapping of VN request onto the shared substrate network

II. RELATED WORK

The VN embedding problem is identical to the existing research on virtual private network assignment in shared provider topology [14] [15]. For a typical VPN request only bandwidth requirements that are represented by traffic matrix without any constraints on its nodes. Algorithms designed by VPN to find paths between source\destination pairs. VPNs usually possesses standard topologies like full mesh and hub-and-spoke [16]. Vineyards in existing system provides improved embedding for standard as well as arbitrary VN request topology [1].

VN assignment is also connected with network testbed mapping problem [17]. In the Emulab testbed [17], the Assign algorithm used which considers bandwidth constraints with constraints on exclusive use of nodes (i.e. different VNs cannot share the substrate node). One of the most important principle of network Virtualization is sharing of substrate nodes and links by multiple VNs [10], this objective must be supported by VN embedding problem. Emulab itself is aligning its resource mapping policies with that of network virtualization [18].

Existing research has been limiting the problem space in different dimensions to reduce the hardness of the VN assignment problem and to enable efficient heuristic, as

1) offline version of the problem considered (i. e. all the VN requests are known in advance) [3] [12].
2) Either node or link requirements are ignored [11] [12]
3) infinite capacity of substrate node and link assumed to obviate from admission control [3] [11]
4) specific VN topologies are focused.

Opposed to the algorithms proposed in [1], all the existing algorithms can be differentiated into two basic stages :
1) assigning virtual nodes using some greedy heuristics (e.g.,assign virtual nodes with higher requirements to substratenodes with more available resources [3], [4]);
2) embedding virtual links onto substrate paths using shortest path algorithms [3] in case of unsplittable flows, or using multicommodity flow algorithms in case of splittable flows [4], [5].

In [19] Lischka proposed a backtracking-based VN embedding algorithm using subgraph isomorphism detection that extensively searches the solution space in a single stage. In [20] Houidi presented a distributed algorithm in which virtual nodes and virtual links are simultaneously mapped without using centralized controller.

It is assumed that the substrate network is always operational. The operations of multiple VNs can be disrupted by failure of the single physical resource. In [21] Rahman extended the ViNEYard model with a fast rerouting strategy that utilizes preallocated backup quota on each physical link to ensure survivability from individual substrate link failures. An integer programming model is proposed in [22] by Kuman to solve the VPN tree computation problem for bandwidth provisioning in VPNs. To obtain approximation algorithms the techniques of randomized rounding for linear programming relaxations was first introduced in [23]. The effect of lookahead on online algorithms have been thoroughly investigated in the algorithm design literature [24]. Mosharaf in [1] proposed a generalized window-based VN embedding mechanism that is based on a fixed window size and prespecified maximum

ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 2, Issue 4, April 2013

waiting periods for VN requests and can extend any existing purely online VN embedding algorithm [1].

III. NETWORK MODEL AND PROBLEM FORMULATION

A. Substrate Network

The substrate network is modeled as a weighted undirected graph $G^S = (N^S, E^S)$, where N^S and E^S is the set of substrate nodes and set of substrate paths. Each substrate node $n^S \in N^S$ is associated with CPU capacity weight value $c(n^S)$ and its location $loc(n^S)$ on the coordinate system [1]. Each substrate link $e^S(i,j) \in E^S$ between two substrate nodes and is associated with the bandwidth capacity weight value $b(e^S)$ denoting total amount of bandwidth [1].

The set of all substrate paths is denoted by P^S and the set of all substrate paths from source node a to destination node b is given by $P^S(a, b)$.

Fig. 1 shows a substrate network, where the numbers over the links represent available bandwidths and the numbers in rectangles represent available CPU resources [1].

B. VN Request

VN Request denoted by $G^V = (N^V, E^V)$ is also a weighted undirected graph, where N^V and E^V is a set of all virtual nodes and sets of all virtual links. The requirements for virtual nodes and virtual links are represented in terms of the attributes of nodes and links of the substrate network. A virtual node $n^V \in N^V$ has CPU capacity requirement $c(n^V)$ and geographical location $loc(n^V)$. A virtual link $e^V \in E^V$ has bandwidth capacity $b(e^V)$ and delay constraint D^V. A nonnegative value D^V is associated with each VN request to represent how far a virtual node $n^V \in N^V$ can be assigned from its preferred location $loc(n^V)$. D^V can be represented as link delay or round-trip time (RTT) from location $loc(n^V)$ [1]. Fig. 1 shows two VN requests with node and link constraints.

C. Measurement of Substrate Network Resources

If the substrate network resources are considered to be finite then new VN request processing reduces the amount of residual substrate network resources. Thus VN request is accepted only if there are sufficient resources to serve it. The concept of stress is used to represent the quantity of resource usage of substrate network. The substrate node stress is denoted as $S_N(n^S)$ and is defined as the total amount of CPU capacity assigned to different virtual nodes hosted on the substrate node $n^S \in N^S$ [1].

$$S_N = \sum_{n^V \to n^S} c(n^V) \quad (1)$$

where $x \to y$ indicates that virtual node x is hosted on substrate node y.

In the same way, substrate link stress $S_N(e^S)$ is defined as the total amount of bandwidth reserved for virtual links whose substrate path pass through substrate link $e^S \in E^S$ [1].

$$S_E(e^S) = \sum_{e^V \to e^S} b(e^V) \quad (2)$$

where $x \to y$ denotes the substrate path of virtual link x pass through the substrate link y.

D. VN Assignment

In virtual network assignment the virtual network topology is mapped at the top of physical network topology based on some requirements. The substrate network decides whether to receive or not the VN request. Once the request is received then it is the duty of the substrate network to allocate resources and decide an appropriate assignment for VN [1] [4] [5]. The assignment of virtual request onto the substrate network is decomposed in two phases:

1) Node Mapping - Each virtual node from the same VN request is assigned to a different substrate node by mapping $M_N : N^V \to N^S$ From virtual nodes to substrate nodes such that for all $n^V, m^V \in N^V$ [1] [5].

$$M_N(n^V) \in N^S$$

$$M_N(m^V) = M_N(n^V) \quad iff\ m^V = n^V$$

subject to

$$c(n^V) \leq R_N(M_N(n^V)) \quad (3a)$$

$$dis(loc(n^V), loc(M_N(n^V))) \leq D^V \quad (3b)$$

where $dis(x,y)$ measures the distance between the locations of two substrate nodes x and y.

In Fig. 1, the first VN request has the node mapping $\{p \to R, q \to W, r \to Q\}$ and the second VN request has $\{s \to P, t \to S, u \to V\}$ [1]. Here two virtual nodes u and q from the different VN request are mapped onto same substrate node W.

2) Link Assignment: Each virtual link is assigned to a substrate path or a set of substrate paths between corresponding substrate node i. e. Mapping $M_E: E^V \to P^S$

ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 2, Issue 4, April 2013

From virtual link to substrate paths such that for all $e^V = (m^V, n^V) \in E^V$ [1].

$$M_E(m^V, n^V) \subseteq P^S(M_N(m^V), M_N(n^V))$$

subject to

$$\sum_{P \in M_N(e^V)} R_E(P) \geq b(e^V) \quad (4)$$

The first VN request in Fig. 1 has assigned the link mapping $\{(p,q) \rightarrow \{(R,S),(S,V),(V,W)\}, (p,r) \rightarrow \{(R,P),(P,Q)\}, (q,r) \rightarrow \{(W,U),(U,T),(T,Q)\}\}$ and the second VN request has link mapping $\{(s,t) \rightarrow \{(P,R),(R,S)\}, (t,u) \rightarrow \{(S,V),(V,W)\}\}$ [1].

E. Objectives

The main purpose of this paper is to propose an algorithm that verifies the integrity and authentication of data packet transmitted from source node to the destination node. A MAC can be thought as a checksum for data passed through an unreliable channel. A sender will typically generate a MAC code by first passing their message into some MAC algorithm. The sender will then send their message M with the MAC(M). The receiver can then generate their own MAC(M) and verify that MAC(M) sent by the receiver matches the MAC(M) they themselves generated. To compute a MAC over the data 'm' using the HMAC function, the following operation is performed [8]:

$$MAC(m) = HMAC_K(m) = H(K \oplus opad; H(K \oplus ipad))$$

HMAC has all of the general properties of a MAC function this means that HMAC is suitable anytime senders and receivers wish to guarantee integrity between sender and receiver.

IV. EXISTING SYSTEM

Modifications to the current architecture of the Internet are difficult due to multi-provider nature of the Internet and competition amongst the providers. To overcome this problem of ossification, the concept of Network Virtualization is introduced. The Network Virtualization approach allow multiple heterogeneous virtual networks to co-exist over shared substrate infrastructure [1] [3]. The existing system provides better co-ordination between node and link mapping phases by presenting ViNEYard algorithms which consist of Deterministic VN embedding (D-VINE), Randomized VN embedding (R-ViNE) and their extensions [1]. In this there is online VN embedding algorithms that map multiple VN requests with node and link constraints. The virtual nodes are mapped onto substrate node in a way that facilitate the mapping of virtual links over a substrate path in subsequent phases is done. The VN embedding deals with mapping of virtual nodes and virtual links onto specific substrate node and substrate path. This mapping is done on the basis of CPU capacity of the node and bandwidth capacity of the link. After mapping transmission of data packets is performed over the network. While transmission security of data packet is not maintained. Due to this when a data packet is transmitted from sender to receiver there may be loss of data or intruders can damage the packet.

A. Augmented Substrate Graph Construction

To construct augmented substrate graph the base substrate network must be extended using location requirements of virtual nodes as the basis for extensions [1]. Creation of augmented substrate graph co-ordinate the node mapping and link mapping phases. As each virtual node $n^V \in N^V$ is associated with constraint *loc(n^V)* on its possible placement, we can create one cluster for each virtual node in the substrate network – each with radius D^V [1]. This cluster is denoted by $\Omega(n^V)$ and we a can say it as the Ω set of the virtual node n^V [1].

$$\Omega(n^V) = \{n^S \in N^S \mid dis(loc(n^V), loc(n^S)) \leq D^V\}$$

For each virtual node $n^V \in N^V$, we create a corresponding meta node $\mu(n^V)$ and connect $\mu(n^V)$ to all the substrate nodes belonging to $\Omega(n^V)$ using meta edges with infinite bandwidth [1].Then all the meta nodes and meta edges are combined with the substrate graph to create an augmented substrate graph.

B. Deterministic Rounding-Based Virtual Network Embedding Algorithm (D-ViNE)

The input of D-ViNE is VN requests and substrate nodes. The online VN requests are mapped one by one onto the substrate network.

1. An algorithm D-ViNE will take VN request $G^V = (N^V, E^V)$
2. Augmented substrate graph $G^{S'} = (N^{S'}, E^{S'})$ is constructed
3. Then VNE_LP_RELAX is solved
4. For all $n^S \in N^S$
5. All substrate nodes are unused thus initially $\varphi : N^S \rightarrow \{0,1\}$, set to zero and when virtual node is mapped to a specific physical node n^S then $\varphi(n^S)$ is set to 1 to ensure that no substrate node is used twice for the same VN request.
6. For each virtual node algorithm checks in Ω set if there is any unmapped substrate node
7. If any of Ω sets is empty, the algorithm stops embedding process and rejects the VN request

ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 2, Issue 4, April 2013

otherwise go to step 8, where the deterministic rounding procedure is initiated.

8. For each substrate node $z \in \Omega(n)$
9. P_Z is calculated as the product of binary variable and total flow passing through meta edge in both directions
10. The substrate node with highest P_Z value is selected. The virtual node n is mapped onto the selected unmapped substrate node z (i.e. $\varphi(z)=0$)
11. Set $\varphi(n^S) = 1$ so as no substrate node is used twice for the same VN request
12. After mapping different substrate node with all the virtual nodes D-ViNE applies the multicommodity flow algorithm to map the virtual links in E^V onto substrate paths.
13. Finally, D-ViNE updates the residual capacities of substrate nodes and links to prepare for the next request.

C. Randomized Rounding-Based Virtual Network Embedding Algorithm (R-ViNE)

R-ViNE is implemented similar as to D-ViNE with the difference that R-ViNE uses randomized rounding and not the deterministic rounding. In R-ViNE, calculated *Pz* values are normalized to restrict them within the range 0-1. For each $z \in \Omega(n)$ the normalized values correspond to probabilities of n being mapped to z. The substrate node $z \in \Omega(n)$ is selected by R-ViNE to map a virtual node n with probability P_Z.

V. PROPOSED SYSTEM

In the proposed system, we introduced a method which performs verification of integrity and authentication of data packet when it is transmitted from source node to the destination node of the substrate network. In the world of open computing and communications it is important to check the integrity of the information transmitted over an insecure channel. Mechanisms that provide integrity checks are based on secret key and takes the form of Message Authentication Code (MACs). HMAC is merely a specific type of MAC function.

It works by using an underlying hash function over a message and a key. It is currently one of the predominant means to ensure that secure data is not corrupted in transit over unsecure channels like the internet.

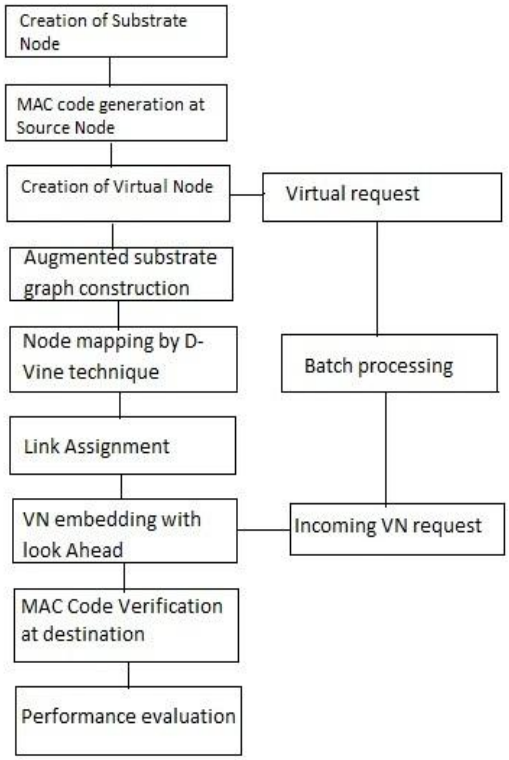

Fig. 2 Proposed System Design

The HMAC function takes the key K and message 'text' as input, and produces [6] [7] :

$$HMAC_K(text) = H(K \oplus opad; H(K \oplus ipad, text))$$

Where H is a cryptographic hash function, K is a secret key padded to the right with extra zeros to the input block size of the hash function, or the hash of the original key if it's longer than that block size, the text is the message to authenticated, $\oplus$ denotes exclusive or (XOR), *opad* is the outer padding (0x5c5c5c….5c5c, one-block-long hexadecimal constant), and *ipad* is the inner padding (0x363636…3636, one-block-long hexadecimal constant).

The following Algorithm shows how HMAC-MD5 is implemented [8] :

1. Generate a key K for HMAC-MD5 keyed-hashing algorithm.
2. Create a MAC object using HMAC-MD5 and initialize with key K.
3. If length of K=Blocksize : Set K_0=K. GO to Step 6.
4. If the length (K) > Blocksize: hash K to obtain L byte string, then append (Blocksize-L) zeros to create Blocksize-byte string K_0 (i.e., K_0=H(K)||00..00). Go to Step 6.
5. If the length (K) < Blocksize : append zeros to the end of K to create a Blocksize-byte string K_0 (e.g.,

ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 2, Issue 4, April 2013

if K is 20 bytes in length and Blocksize=64, K will be appended with 44 zero bytes x'00')

6. Exclusive-Or K_0 with ipad to produce a Blocksize-byte string : $K_0 \oplus$ ipad.
7. Append the stream of data 'text' to the string resulting from step 6 : ($K_0 \oplus$ ipad) || text.
8. Apply H to the stream generated by step 7 : H(($K_0 \oplus$ ipad) || text).
9. Exclusive-Or K_0 with opad : $K_0 \oplus$ opad.
10. Append the result from step 8 and 9 : ($K_0 \oplus$ opad)|| H(($K_0 \oplus$ ipad) || text).
11. Apply H to the result from step 10 : H (($K_0 \oplus$ opad)|| H(($K_0 \oplus$ ipad) || text)).

VI. PERFORMANCE ANALYSIS

In this section we describe a performance analysis of the VN request and efficiency of the system. The dataset contains the values of time obtained and the number of virtual request arriving to the substrate node at a given time. During the execution of the first VN request multiple VN requests are generated as the background processing. All the VN requests try to map with the given substrate nodes. Thus the substrate node decides whether to accept the request or not. The arrival of multiple VN requests also creates traffic at the substrate node which causes traffic collision. To avoid this collision in our proposed system, better coordination between node and link mapping phases is done by applying window period to each VN request. Each VN request is processed in a queue means one request at a time. The graph below shows the performance of VN requests over given time obtained.

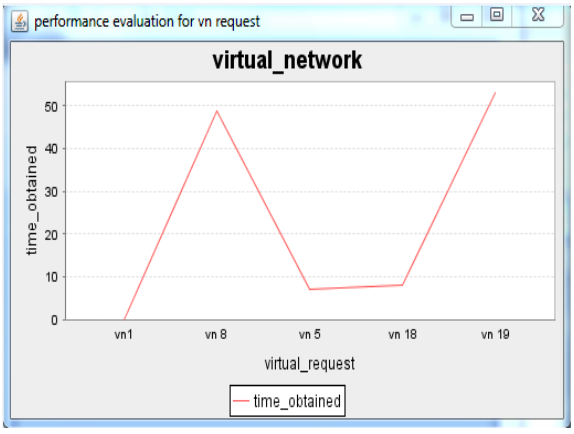

Fig 2 VN request performance

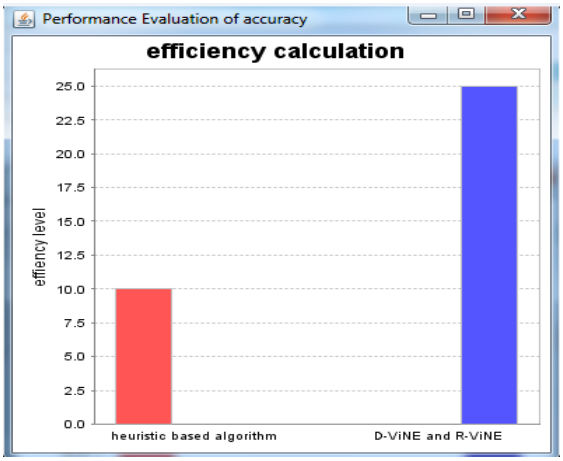

Fig. 3 Performance Evaluation of Accuracy

VII. CONCLUSION

Network Virtualization is an important approach to overcome the problem of ossification. Coordinated node and link mapping stages of VN embedding process increase the solution space and improves quality of embedding. In Network Virtualization environment, each virtual network (VN) is separated from the others and a set of virtual network shares the resources of a common physical network. In various Internet applications and protocols the standard approaches like cryptographic hash functions are used for the authentication of data. When data packet is transmitted the integrity is verified which improves security of information. For the purpose of improving security and reduce the loss of data or information during the process of transmission we use an algorithm that encodes the data and key at sender side and decodes it at the receiver side. The focus of the paper is in the authentication of the data being transmitted in the network and Integrity of the data is also verified.

ACKNOWLEDGMENT

I wish to express deep sense of gratitude to my guide Prof. V. P. Kshirsagar for his valuable and firm suggestions, guidance, encouragement and constant support throughout this work without which it would not be possible to do this work. He took deep interest in checking the minute details of the paper and guided throughout the same. He has been constant source of inspiration.

REFERENCES

[1] Mosharaf Choudhary, Muntasir Raihan Rahman, Raouf Boutaba "ViNEYard : Virtual Network Embedding Algorithms With Coordinated Node and Link Mapping" in Proc IEEE/ACM Transactions on Networking, 2012.

ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 2, Issue 4, April 2013

[2] T. Anderson, L. Peterson, S. Shenker and J. Turner "Overcoming the internet impasse through network Virtualization" Computer, vol. 38, no. 4, pp. 34–41, 2005.
[3] Yong Zhu, Mostafa Ammar "Algorithms For Assignment of Substrate Network Resources to Virtual Network Components" In Proc. IEEE INFOCOM, 2006.
[4] Minlan Yu, Ying Yi, Jennifer Rexford, Mung Chiang "Rethinking Virtual Network Embedding: Substrate Support for Path Splitting and Migration" Comput. Commun, Rev., vol. 38, no. 2, pp. 17–29, Apr. 2008.
[5] W. Szeto, Y. Iraqi and R. Boutaba "A multi–commodity flow based approach to virtual network resource allocation", in Proc. IEEE GLOBECOM, 2003.
[6] Abdeltouab Belbekkouche, Md. Mahmud Hasan, Ahmed Karmouch "Resource Disovery and Allocation in Network Virtualization " in Proc IEEE Communications, 2012.
[7] Muntasir Raihan Rahman, Isaam Aib, Rouf Boutaba "Survivable Virtual Network Embedding" IFIP NETWORKING, pp. 40-52, 2010.
[8] Mihir Bellare, Ran Cannetti, Hugo Krawczyk "Keying Hash Functions for Message Authentication" Advances in Cryptology - Crypto 96 Proceedings, Lecture Notes in Computer Science Vol. 1109, N. Koblitz ed., Springer-Verlag, 1996.
[9] Mihir Bellare, Ran Cannetti, Hugo Krawczyk "Message Authentication using Hash Functions - The HMAC Construction" Appears in RSA Laboratories' CryptoBytes, Vol. 2, No. 1, Spring 1996.
[10] National Institute of Standards and Technology, The Keyed-Hash Message Authentication Code (HMAC), Federal Information Processing Standards FIPS PUB 198-1, July 2008.
[11] N. M. M. K. Chowdhury and R. Boutaba , "A survey of network virtualization" Compu. Netw., vol 54, no. 5, pp 862-876, 2010.
[12] J. Fan and M. Ammar, "Dynamic topology configuration in service overlays networks - A Study of reconfiguration policies" in Proc. IEEE INFOCOM, 2006, pp 1-12.
[13] J. Lu and J. Turner "Efficient mapping of virtual networks onto shared substrate", Washington University, Seattle, WA, Tech. Rep. WUCSE, 2006.
[14] A. Gupta, J. M. Kleinbarg, A. Kumar, R. Rastogi and B. Yener "Provisioning a virtual private network : A network design problem for multicommodity flow" in Proc. ACM STOC, 2001.
[15] N.G. Duffield, P. Goyal, A. Greenberg, P. Mishra, K. K. Ramkrishanan and J. V. van der Merwe, "Resource Management with hoses: Point-to-cloud services for virtual private network", IEEE/ACM Trans., 2002.
[16] S. Raghunath, K.K. Ramkrishnan, S. Kalyanaraman and C. Chase, " Measurement based characterization and provisioning of IP VPNs," in Proc. ACM IMC, 2004.
[17] R. Ricci, C. Alfeld and J. Lepreau, "A solver for the testbed mapping problem," Comput. Commun., Apr 2003.
[18] M. Hibler, R.Ricci, L. Stoller, J.Duerig, S. Guruprasad, T. Stack, K. Webb and J.Lepreau, "Large-scale virtualization in the Emulab network testbed" in Proc. USENIX ATC, 2008.
[19] J. Lischka and H. Karl, "A virtual network mapping algorithm based on subgraph isomorphism detection", in Proc. ACM VISA, 2009.
[20] I. Houidi, W. Louati, and D. Zeghlache, "A distributed virtual network mapping algorithm", in Proc. IEEE ICC, 2008.
[21] M.R. Rahman, I. Aib, R. Boutaba, "Survivable virtual network embedding", in Proc. IFIP, 2010.
[22] A. Kumar, R.Rastogi, A. Silberschatz and B. Yener, "Algorithms for provisioning virtual private networks in the hose model," *IEEE/ACM Trans. Netw.*, Aug. 2002
[23] P. Raghavan and C. D. Tompson, "Randomized rounding: A technique for provably good algorithms and algorithmic proofs," *Combinatorica*, 1987
[24] S. Albers, "A competitive analysis of the list update problem with lookahead," *Theor. Comput. Sci.*, vol. 197, no. 1–2, pp. 95–109, 1998.
[25] Sandip Kankal, Prof. V. P. Kshirsagar, Prof. Harpreet Singh Soch " An improved algorithm for verification of integrity and authentication of data in virtual networks.", in Proc. , ICITEC, 2013.

Biography

Prof. V. P. Kshirsagar has been working as a Professor in Computer Science and Engineering department since last 17 years. Author has received his B.E. degree in CSE, M.E. degree in CSE and currently doing his Ph.D in the area of Networking.

Sandip Kankal received his B.E. degree in Information Technology from MGMs. JNEC, Aurangabad, Dr. BAMU University, India, in 2009 and pursuing M.E. degree in computer science and engineering from Government College of Engineering Aurangabad, India. His research interests include the area of Networking.