INVITED PAPER Special Section on Technologies and Architectures for Improving Scalability, Reliability, and Robustness for Future Information Networks

ZNA: A Six-Layer Network Architecture for New Generation Networks —— Focusing on the Session Layer, the Network Layer, and Cross-Layer Cooperation ——

Fumio TERAOKA(1), Sho KANEMARU(1+)*, Members, Kazuma YONEMURA(1††), Motoki IDE(1†††)*, Shinji KAWAGUCHI(1), Nonmembers, and Kunitake KANEKO(1††††), Member

SUMMARY Using “clean-slate approach” to redesign the Internet has attracted considerable attention. ZNA (Z Network Architecture) is one of clean-slate network architectures based on the layered model. The major features of ZNA are as follows: (1) introducing the session layer to provide the applications with sophisticated communication services, (2) employing inter-node cross-layer cooperation to adapt to the dynamically changing network conditions, (3) splitting the node identifier and the node locator for mobility, multi-homing, and heterogeneity of network layer protocols, (4) splitting the data plane and the control plane for high manageability, and (5) introducing a recursive layered model to support network virtualization. This paper focuses on the first three topics as well as the basic design of ZNA.

key words: clean-slate approach, session layer, ID/Locator split, cross-layer cooperation

1. Introduction

The basic architecture of the Internet was designed from the late 1970’s to the beginning 1980’s. As widely known, the Internet is based on the five-layer architecture. Since then, several improvements such as security, mobility and label switching have been added to the basic five-layer architecture and the today’s Internet has formed. However, such newly added functions do not fit into the five-layer architecture. For example, IPSec and Mobile IP are called layer-3.5 (L3.5) protocols and MPLS is called a layer-2.5 (L2.5) protocol. There are strong opinions that the progress of the Internet will be ended in failure if we continue to add new functions that do not fit into the basic Internet architecture.

The belief that the Internet should be redesigned from scratch has been growing in 2000’s. Such an approach is called the clean slate approach. There were several research projects that aimed at redesigning network architectures based on the clean slate approach, e.g., in USA, the New Arch Project [1] supported by DoD (Department of Defense) in 2002–2003, the 100 × 100 Clean Slate Project [2] in 2003–2005, the FIND (Future Internet Design) Project [3] supported by NSF (National Science Foundation) in 2006–2013, and the FIA (Future Internet Architecture) Project [4] supported by NSF in 2010. In addition, the GENI (Global Environment for Network Innovations) Project [5] started in 2007 with the support of NSF. GENI is not the future Internet, but it focuses on testbeds for developing new technologies based on new network architectures. In EU, the ICT (Information and Communication Technologies) area of the FP7 (Seventh Framework Programme) Project [6] in 2007–2013, was promoting researches on the future network such as the Trilogy Project [7] and the 4WARD Project [8]. The ICT area also employed the clean slate approach. In Japan, the AKARI Project [9] in NICT (National Institute of Information and Communications Technology) aimed at designing a New Generation Network (NwGN) in the clean slate approach in 2006–2010.

As a member of the AKARI project, one of the authors proposed a six-layer network architecture called ZNA (Z Network Architecture) for NwGN. The major features of ZNA are as follows: (1) introducing the session layer to provide the application layer with sophisticated communication services, (2) employing cross-layer cooperation not only in the same node but also in different nodes to adapt to the dynamically changing network conditions, (3) splitting the node identifier and the node locator for mobility, multi-homing, and heterogeneity of network layer protocols, (4) splitting the data plane and the control plane for manageability, and (5) introducing a recursive layered model to support network virtualization.

Among those features, this paper introduces the first three features as well as the basic design of ZNA [10]. First, this paper discusses the reasons why ZNA adopts the layered architecture with cross-layer cooperation although there are several proposals based on non-layered architecture. Second, a new layer, the session layer, inserted between the application layer and the transport layer is introduced. The session layer provides the application layer with three kinds of sophisticated communication services: the bundled path, the spatially-spliced path and the temporally-spliced path [11], [12]. Third, a network layer protocol called ZNP (Z Network Protocol) is introduced. ZNP supports mobility and multi-

† The authors are with Keio University, Yokohama-shi, 223-8522 Japan.
*Presently, with NTT Network Service Systems Laboratories.
**Presently, with Nomura Research Institute.
***Presently, with DWANGO Co., Ltd.
a) E-mail: tera@ics.keio.ac.jp
b) E-mail: satsuma@inl.ics.keio.ac.jp
c) E-mail: shallot@inl.ics.keio.ac.jp
d) E-mail: alfy@inl.ics.keio.ac.jp
e) E-mail: kaneko@ics.keio.ac.jp
DOI: 10.1587/transcom.E97.B.2583
homing by splitting the node ID and the node locator [13], [14]. ZNP also allows heterogeneous network layer protocols to coexist based on the concept internetworking with a common ID space. Fourth, CLINEX (Cross-Layer and Inter-Node information Exchange) is introduced. CLINEX enables cooperation between layers not only in the same node but also in different nodes so that nodes can adapt to the dynamically changing network conditions [15].

2. Z Network Architecture

2.1 Consideration on Layered Model

The OSI (Open Systems Interconnection) reference model defines the seven-layer model. It was standardized as ISO7498 in 1984 and was revised in 1994 [16]. In the layered model, the functions that realize networking are divided into several layers. The details of a layer are concealed from the general fine-grained building blocks called services. The Internet architecture is composed of five layers.

In the layered model, however, the layering structure is static and cannot be dynamically changed. To relax this restriction, several new network architectures were proposed that were not based on the layered model such as RBA (Role-Based Architecture) [17] proposed in the New Arch Project and SILO [18] proposed in FIND. In RBA, a function unit is called the role. Roles can be more freely interconnected than the layers. However, as the paper [17] mentioned, RBA focuses on redesigning the transport layer and the network layer, not on redesigning the entire network architecture. The approach of SILO is the middle of the current static layered structure and non-layered structure such as RBA. In SILO, a function is divided into several fine-grained building blocks called the services. The just-in-time protocol suite is composed by appropriate services according to the requirements of the application. In addition, the control API is defined other than the data API to enable cross-layer cooperation. However, since the order of applying the services is pre-defined among services, the just-in-time protocol suite might result in static structures. Although RBA and SILO are very attractive in terms of flexibility, both of them focus on the structure of the layers higher than the network layer. In addition, it seems that both of them have rather high overhead and it is very difficult to implement RBA or SILO in practice.

Network architecture is not free from the physical configuration of the network. A lot of functions are required for the communication between applications. It is quite natural to build the functions required for endpoint-to-endpoint communication (the transport layer) above the functions required for node-to-node communication (the network layer), which exist above the functions required for point-to-point communication (the link layer). Thus, it is quite natural to employ the layered model for the network architecture of the NwGN. A lot of functions must be defined in each layer to provide the services to the upper layer. In most cases, it is adequate to pre-define the order of applying the functions in a layer. Whether a function is applied or not should be specified by parameters.

2.2 Consideration on Information Centric Networking

ICN (Information Centric Networking) [19]–[21] has attracted considerable attention as a new paradigm for NwGN. One of the most attractive ICN mechanisms is NDN (Named Data Networking) [20]. NDN changes the communication model from “host-based” to “content-based”. In the current Internet, when a client wants to retrieve content such as a video in YouTube, the client eventually specifies the node via URL that holds the requested content even if the client searches for the content through a search engine. In NDN, the client sends an “Interest packet” that contains the location-independent name of the content the client wants to retrieve. The Interest packet is routed based on the content name. When the Interest packet encounters in-network cache or reaches the node that holds the content, the content is transferred in a “Data packet” to the client on the reverse path of the Interest packet.

NDN tires to realize not only retrieval of stored content such as videos in YouTube but also live streaming such as VoIP. However, NDN is not a “one-fits-all” communication mechanism even through NDN could realize any types of applications. In case of VoIP, e.g., it is quite natural to establish a connection between the communicating nodes as Skype rather than to retrieve each voice packet by specifying the name of each voice packet in NDN.

Even in the case of stored content retrieval, layered model is superior to NDN in terms of scalability and performance. In NDN, name-based routing information must be announced throughout the network. If the same content is stored in multiple nodes for fault tolerance, there must be multiple routing entries to the content but these routing entries cannot be aggregated. Thus, name-based routing does not scale. After finding the content, the Data packet is processed by each node on the reverse path of the Interest packet. This mechanism cannot achieve high throughput for large content.

In the layered model, a location-independent content name can be resolved to the node that holds the content in the application layer. The content can be transferred to the client by TCP or multi-path TCP, which may achieve higher throughput than NDN. Routing information other than the ordinary network address based routing information is not necessary to be announced. It is possible to employ in-network cache in the session layer (see Sect. 3.4).

2.3 Consideration on Cross-Layer Cooperation

Major advantages of the layered model are (1) abstraction of functions of lower layers and (2) independence of each layer. Due to abstractions of lower layers, a protocol does not need to consider how the services provided by lower layers are realized, i.e., a protocol can use the services provided
by lower layers through the SAP (Service Access Point) located at the boundary of two adjacent layers. Due to independence of each layer, functions of each layer can be designed or modified independently as long as the services provided through the SAP remain unchanged.

Protocol layering makes it easier to develop and deploy new functions in the network, however, it has several disadvantages. The network conditions such as congestion state, routing state, and failure state of nodes or links are dynamically changing in the current Internet. In some cases, it would be better for a protocol to use control information in other layers to adapt to the dynamically changing network conditions. A typical example is to adapt to wireless communication environment in which the link layer cooperates with upper layers to achieve fast handover [22].

2.4 Consideration on ID/Locator Split

In the current Internet, the IP address has two meanings: the identifier (ID) of a node and the locator of a node. Such duality of the IP address makes it difficult to support mobility and multi-homing. To support mobility and multi-homing, a lot of protocols based on ID/Locator split approach have been proposed. ZNA also adopts the ID/Locator split approach.

Furthermore, ZNA proposes a concept internetworking with a common ID space. In the current Internet, only IP (Internet Protocol) is the network layer protocol. In other words, all devices connected to the Internet must speak IP. In NwGN, not only powerful devices such as computers but also powerless devices such as sensors will be connected to the network. Powerless devices may adopt a simple network layer protocol while powerful devices may speak ZNP (see Sect. 4) as the network layer protocol. Thus, we assume that multiple network layer protocols coexist in NwGN. To enable communication between nodes that use different network layer protocols, ZNA employs a single node ID space for all nodes connected to the network even if each node uses a different network layer protocol. ZNA makes it possible to communicate with another node by specifying the node ID even if the node uses a different network layer protocol.

2.5 Basic Architecture of ZNA

As a result of the consideration above, ZNA adopts the layered model with cross-layer cooperation and ID/Locator split as depicted in Fig. 1. ZNA is composed of six layers. Each layer is named according to the OSI reference model. As the figure shows, the data plane and the control plane are separate. The data plane contains the functions for data communication as depicted in Fig. 2 while the control plane contains the functions for network control signaling as depicted in Fig. 3. Abstract Entities (AEs), Inter-Layer System (ILS), and Inter-Node System (INS) are employed for inter-node cross-layer cooperation (see Sect. 5). The session layer (L5) is newly introduced to provide the application layer with sophisticated services (see Sect. 3).

In Fig. 2, the shaded rectangles are services provided by a layer to its upper layer while the clear rectangles are functions in each layer. Since ZNA employs ID/Locator split, the network layer is divided into two sub-layers: the ID sub-layer and the forwarding sub-layer. In the layered model, a service in a layer is realized by some functions in the same layer and some services provided by the lower layer. An example is shown below. The L5-reliable-path service provides the application layer with reliable communication service. In case that the bundled-path in the session layer (see Sect. 3.3) is used, the L5-reliable-path service is realized by the bundled-path function in the session layer and the L4-reliable-path service provided by the transport layer (L4). The L4-reliable-path service is realized by the sequence control function and the reliability control function in the transport layer, and the L3-BE-path (L3 best effort path) service provided by the network layer (L3). The L3-BE-path service is realized by the ID/Locator mapping function and the datagram forwarding function in the network layer, and the L2-link service provided by the link layer (L2), and so on.

The rectangles in Fig. 3 are signaling functions in
each layer. To provide the L5-reliable-path service on the bundled-path function, the following signaling functions are used: the L5-reliable-path control signaling, the bundled-path control signaling function, and the reception status feedback function in the session layer; the L4-reliable-path control signaling function and the reception status feedback function in the transport layer; the ID/Locator mapping control signaling function in the link layer and so on.

3. Sophisticated Services Provided by Session Layer

3.1 Roles of Session Layer

The transport layer of the current Internet provides applications with only two kinds of communication services: reliable byte stream and unreliable datagram. These two services are too simple to build practical network applications. To fill the gap between the services provided by the transport layer and the services required by practical applications, application programs must implement missing functions by themselves. The functions implemented in an application program cannot be shared with other application programs.

There is another approach to fill the gap, in which new functions are added in the transport layer to provide the applications with required services. In this approach, applications can utilize the improved transport services. As a result, however, this approach makes implementation of the transport layer very complicated.

ZNA introduces the session layer between the application layer and the transport layer. The session layer provides the application layer with sophisticated communication services called the L5-paths by combining the transport layer paths (L4-paths). There are four types of L5-paths: the regular path, the bundled path, the spatially-spliced path and the temporally-spliced path. By dividing basic communication mechanisms (the transport layer) from sophisticated services (the session layer), a new service can be added by modifying the session layer without modifying the transport layer.

3.2 Regular Path

The regular path is a L5-path that is mapped to a single L4-path. It provides the application layer with the same service as the current transport layer provides.

3.3 Bundled Path

The bundled path is a L5-path composed of two or more end-to-end L4-paths as shown in Fig. 4. A L5-path with bandwidth aggregation is realized if two or more L4-paths are used simultaneously. A L5-path with fault tolerance is realized if a single L4-path is used as the primary path and others are reserved as secondary paths. If the primary path becomes unavailable, one of the secondary paths is engaged as the primary path. Thus, the bundled path is a model of multi-path communication.

In the current Internet, SCTP [23] provides the same services as the bundled path provides. The design of SCTP is very complicated because it manipulates simple end-to-end paths and the association composed of several paths in the same layer.

3.4 Spatially-Spliced Path

In the current Internet, most subscribers are connected to the global Internet through a NAT (Network Address Translation) box, which translates private addresses used in the subscriber network into global addresses. Originally, a NAT box was introduced as a means to cope with IPv4 address exhaustion. There are some negative opinions on existence of NAT boxes because it violates the end-to-end principle while there are several advantages on their existence from viewpoint of network operation and management. A NAT box also serves as a firewall or it can apply local policy. ZNA assumes that middle boxes such as NAT boxes still exist in NwGN from viewpoint of network operation and management. ZNA aims at a network architecture that allows existence of middle boxes, not excludes them.

Therefore, ZNA introduces the spatially-spliced path, a model of handling middle boxes, that splices two L4-paths divided spatially as shown in Fig. 5. In Fig. 5, there are two L4-paths divided by the intermediate node called the splicer. The session layer of the splicer splices the two L4-paths to compose the end-to-end L5-path. In the current Internet, if
must be larger. Thus, DTN (Delay Tolerant Networking) is applied when communication disruption occurs frequently, communication delay is unavoidable, and the application must take communication disruption into account. If communication disruption is an ordinary case, a communication mechanism must be designed to handle disruption in the Internet. On the other hand, there are a lot of proposals to support node mobility in the Internet. Based on such an assumption, there are a lot of proposals to support node mobility in the Internet. On the other hand, there are research proposals that regard mobile nodes as an ordinary case and regard stationary nodes as a special case [24]. If node mobility is an ordinary case, a communication mechanism must take communication disruption into account. If communication disruption occurs frequently, communication delay must be larger. Thus, DTN (Delay/Disruption Tolerant Networking) technologies must be introduced.

Therefore, ZNA introduces the temporally-spliced path, a model of handling communication disruption, that splices several L4-paths not existing at the same time as shown in Fig.6. At time $t_1$ in Fig.6(a), a L4-path is established between the leftmost node and the splicer but there is no L4-path between the splicer and the rightmost node. At time $t_2$ in Fig.6(b), the situation is contrary. In the temporally-spliced path, the splicer forwards packets by buffering them.

3.5 Temporally-Spliced Path

In the current Internet, most end nodes are stationary and node mobility is a special case. Based on such an assumption, there are a lot of proposals to support node mobility in the Internet. On the other hand, there are research proposals that regard mobile nodes as an ordinary case and regard stationary nodes as a special case [24]. If node mobility is an ordinary case, a communication mechanism must take communication disruption into account. If communication disruption occurs frequently, communication delay must be larger. Thus, DTN (Delay/Disruption Tolerant Networking) technologies must be introduced.

Therefore, ZNA introduces the temporally-spliced path, a model of handling communication disruption, that splices several L4-paths not existing at the same time as shown in Fig.6. At time $t_1$ in Fig.6(a), a L4-path is established between the leftmost node and the splicer but there is no L4-path between the splicer and the rightmost node. At time $t_2$ in Fig.6(b), the situation is contrary. In the temporally-spliced path, the splicer forwards packets by buffering them.

3.6 Session Layer API

There are several approaches to designing API. One of the approaches is that the API provides application programmers with high-level functions such as “establish a bundled path for bandwidth aggregation to the specified node.” In this approach, the session layer controls the path establishment request from the initiator. Similar to the TCP connection establishment procedure, the initiator calls the socket() function and the responder calls the accept() function. Note that the connect() function is not used in this approach.

In ZNA, the end point of communication path is also called the socket. The socket in ZNA is composed of a port number and an IP address. The IP address specifies the node and the port number specifies the SAP through which the transport layer services are provided to the application layer.

In ZNA, the end point of communication path is also called the socket. The socket in ZNA is composed of a port number and a node ID. Since ZNA employs an ID/Locator split approach, a node is specified by its node ID, not its locator. The port number specifies the SAP through which the session layer services are provided to the application layer.

The L4-socket is composed of a port number and a locator. Since ZNA allows coexistence of various network layer (L3) protocols, the format of the locator in the L4-socket depends on the L3 protocol the node uses. The port number of the L4-socket is the same as that of the L5-socket except for the bundled path. In the bundled path, a single L5-socket might be mapped to several L4-sockets.

3.7 Bundled Path Establishment Procedure

Due to space limitation, this paper shows only bundled path establishment procedure in Fig.7. In the figure, “the initiator” is the node that actively starts path establishment procedure while “the responder” is the node that waits for path establishment request from the initiator. Similar to the TCP connection establishment procedure, the initiator calls the socket() function and the responder calls the accept() function. Note that the connect() function is not used in this approach.

In Fig.7, “ID(i),ID(r)” means that the source node identifier is initiator’s identifier and the destination node identifier is responder’s identifier. When the application
in the initiator calls `connect()`, three messages (L5/L4-SYN, L5/L4-SYN+ACK, and L5/L4-ACK) are exchanged between the session layers in both nodes (Fig. 7(1)). In this procedure, both nodes exchange their locators although Fig. 7 does not show this exchange. As a result, a L5-path containing a single L4-path is established (Fig. 7(2)). After this, data can be transmitted by `send()` and `recv()`. Next, both nodes obtain the locators of the peer node by `getsockopt()` (Fig. 7(3)). Next, the applications on both nodes negotiate how many additional L4-paths should be established (Fig. 7(4)). Suppose that the applications decide that one more L4-path is necessary. To establish another L4-path, the initiator calls `connect4path()` while the responder calls `accept4path()`.

Then, three messages (L4-SYN, L4-SYN+ACK, and L4-ACK) are exchanged between the transport layers in both nodes (Fig. 7(5)). As a result, the second L4-path is incorporated into the L5-path (Fig. 7(6)). As described in Sect. 3.3, the bundled path has two types: bandwidth aggregation and fault tolerance. The applications in both nodes call `setsockopt()` to inform the session layer which type is used (Fig. 7(7)).

3.8 Details and Current Status of the Session Layer

See the paper [11] for the details of the session layer API and the communication procedures of the L5-paths. Currently, the session layer has been implemented in the user space of Linux (Ubuntu 10.04), which realizes the bundled path and the spatially-splitted path by using the normal socket API [12]. In addition, the bundled path has been implemented in the kernel space of Linux-2.6.38 on top of TCP.

Some performance evaluations about the bundled path implemented in the kernel space are shown below. Note that the bundled path establishment procedure in the implementation is slightly different from Fig. 7 because it is implemented on the existing TCP/IP stack, not on the ZNA stack. When the initiator establishes a bundled path, it calls the following system calls: `socket()`, `connect()`, `getsockopt()`, `connect4path()` and `setsockopt()`. The establishment time of a bundled path at the initiator is defined as the period from the time when the initiator calls `socket()` and the time when `setsockopt()` returns. As a result of our measurement, the establishment time of a bundled path at the initiator is 2,211 μsec + 3 \times RTT, where RTT is the round trip time between the initiator and the responder.

Next, preliminary results of throughput measurement are shown below. When a bundled path has two L4 connections, the throughput is approximately 1.5 times that of a single L4 connection. When a bundled path has three L4 connections, the throughput is approximately 1.9 times that of a single L4 connection. Since the throughput is not so good, it is necessary to improve our bundled path implementation.

4. Mobility and Multi-Homing Support by ID/Locator Split in Network Layer

4.1 Requirements to ID/Locator Split Protocols

We defined the following requirements to the network layer protocol based on ID/Locator split in NwGN: (1) support of network layer (L3) protocol heterogeneity, (2) scalability of ID/Locator mapping system, (3) independence of mapping information management, and (4) avoidance of locator leakage beyond the administrative boundary. In NwGN, several L3 protocols such as ZNP, IPv4, IPv6, and other protocols will coexist while the current Internet uses only IPv4 and/or IPv6. Therefore, the requirement-(1) must be satisfied. Nodes that use different L3 protocols must be able to communicate with each other by specifying the identifier of the target node. The requirement-(2) is obvious. A protocol based on ID/Locator split must have an ID/Locator mapping mechanism. To support a huge number of nodes, the mapping system must be scalable. The requirements-(3) and (4) are related to administration viewpoint. The requirement-(3) means that the mapping information of a node must be managed in the administrative domain to which the node belongs (the home domain) in terms of authentication of mapping information when the node is temporarily connected to a foreign domain. If the mapping information is managed in administrative domains other than the home domain, it is very difficult to authenticate the mapping information. The requirement-(4) means that the locator information in an administrative domain must not be registered with the mapping system in other administrative domains. This avoids leakage of the topology information of an administrative domain to the outside.

There are several network layer protocols that supports mobility and multi-homing by ID/Locator split. These protocols can be divided into two approaches. An approach aims at improving the current Internet [25]–[30]. The other approach aims at NwGN [31]–[35]. None of the proposals described above satisfy the four requirements. ZNP was designed so that it satisfies the requirements.
4.2 Assumed Network Structure

The current Internet is composed of the backbone (i.e., a set of network providers) and a large number of subscriber networks. Some proposals assume that the future Internet has the following structures: there are a lot of domains each of which may use a different locator type, and these domains are interconnected and compose a random structure or a multi-level tree structure. However, it is very difficult to manage the network that has such structures. Although there will exist such network structures during a transition period, we assume that the final structure of NwGN converges upon the structure shown in Fig. 8. It is assumed that transition of L3 protocols will be carried out as follows. First, a few network providers start operation of a new L3 protocol in addition to the current L3 protocols. If the new L3 protocol is superior to other L3 protocols from several aspects such as functionality, scalability, and manageability, the number of network providers that adopt the new L3 protocol increases. As a result, most network providers adopt the new L3 protocol and stop supporting the current L3 protocols. Finally, most subscriber networks also adopt the new L3 protocol and stop using the current L3 protocols. However, a few network providers and subscriber networks might continue to use the current L3 protocols. In Fig. 8, similar to the current Internet, there is the backbone network composed of several providers and a large number of edge (subscriber) networks. The backbone network uses ZNP and the edge networks may use ZNP or other L3 protocols.

From locator type viewpoint, NwGN is divided into two spaces: the Universal Locator Space (ULoc-Space) in which the locator of ZNP is used and the Local Locator Space (LLoc-Space) in which legacy L3 protocols such as IPv4/v6 are used. From locator scope viewpoint, the ULoc-Space is further divided into two kinds of networks: the Global Universal Network (GU-Net) in which the scope of the universal locator is the whole future Internet and the Private Universal Network (PU-Net) in which the scope of the universal locator is the inside of the network. Some edge networks belong to the GU-Net while some edge networks belong to the PU-Net similar to the current subscriber networks that connect to the Internet backbone via NAT-like devices. We assume that NAT-like devices still remain in NwGN because there can be administrators who do not want to disclose the network topology to the outside. It is important to design an L3 protocol that permits existence of NAT-like devices, not to prohibit existence of them. A network belonging to the LLoc-Space is called the L-Net. An L-Net is connected to the GU-Net via the gateways that have protocol conversion function.

4.3 Name and ID

In ZNP, the node name is a human readable character string and its syntax is the same as that of the FQDN in the current Internet such as node.x.keio.jp. The ID is fixed length binary data. The format of the ID is shown in Fig. 9. The ID is composed of three parts: the Registry Identifier, the Organization Identifier, and the Node Identifier. The Node Identifier is assigned by an organization (e.g., Keio Univ.) to which the node belongs. The Organization Identifier is assigned by a registry (e.g., JPNIC and APNIC in the current Internet) by which the organization is managed. The Registry Identifier is assigned by an authority such as ICANN in the current Internet.

A node is assigned a locator by the network to which the node is connected. For example, if a node is connected to the ULoc-Space, the node is assigned a universal locator (i.e., a ZNP locator) while if it is connected to the LLoc-Space, it is assigned a local locator such as the IPv4 address. To support L3 protocol heterogeneity, ZNP introduces the protocol conversion function on the gateways that connect the ULoc-Space and the LLoc-Space.

4.4 Mapping System Structure

ZNP has two kinds of mapping systems. One is the Name Mapping System that maps the name to the ID. Another is the ID Mapping System that maps the ID to the locator. Thus, ZNP achieves “Internetworking with a Common ID Space.” A node can communicate with another node as long as it knows the ID of the target node even when the target node moves (node mobility), the network to which the target node is connected moves (network mobility), the target node has multiple interfaces (multi-homing), or various L3 protocols coexist (L3 protocol heterogeneity, the requirement-(1)).

The Name Mapping System (NMS) maps the node name to the corresponding ID. The ID Mapping System (IMS) maps the ID of the node to the corresponding locator. The NMS is composed of the Name Mapping Agents (NMAs) and the IMS is composed of the ID Mapping Agents (IMAs) as shown in Fig. 10. The NMA and the IMA compose tree structures rooted by the root NMA. Similar to the
current DNS, the hierarchy of the NMS is based on the hierarchical structure of the name. The hierarchy of the IMS is based on the hierarchical structure of the ID shown in Fig. 9. An NMA holds the locators of the lower level NMAs and an IMA holds the locators of the lower level IMAs. In addition, the NMA of a domain holds the locator of the IMA that manages the same domain.

Figure 10 shows an example of tree structures of the NMS and IMS that manage a domain jp and its two subdomains unet.jp and pnet.jp. Suppose that unet.jp uses the GU-Loc (i.e., the global scope ZNP locator) while pnet.jp uses the PU-Loc (i.e., the private scope ZNP locator). For pnet.jp, there are two kinds of NMAs and IMAs: the global NMA/IMA and the local NMA/IMA (L-NMA/L-IMA). The global NMA and IMA are located in the GU-Net for the queries sent from the GU-Net. The L-NMA and the L-IMA are located in the Pu-Net for the queries sent within the Pu-Net. In this example, there are one or more NAT-like gateways at the boundary of jp and pnet.jp because pnet.jp is a Pu-Net. The global IMA of pnet.jp holds the mappings between the IDs of the nodes in pnet.jp and the GU-Loc of the gateway. The L-IMA of pnet.jp manages the mappings between the IDs of the nodes in pnet.jp and their private locators.

As described above, name-to-ID mappings and ID-to-locator mappings are managed in a distributed manner, i.e., each name-to-ID mapping and each ID-to-locator mapping are managed by the mapping server of the subdomain to which the node belongs. Thus, ZNP achieves the scalability of the mapping systems (the requirement-(2)) and independence of mapping information management (the requirement-(3)). In addition, by introducing the L-NMA and the L-IMA, ZNP avoids leakage of private or local locators beyond the administrative boundary (the requirement-(4)).

4.5 Signaling Examples

There are several cases in signaling for locator resolution. Due to space limitation, this paper introduces one of the simplest cases: the requesting node and the target node belong to the GU-Net and they are currently connected to the GU-Net.

Suppose that the node Node_X.unet1.jp (Node_X) wants to communicate with the node Node_Y.unet2.jp (Node_Y). Both end nodes exist in the GU-Net (i.e., both end nodes use the universal ZNP locator). This procedure is shown in Fig. 11. Before the communication, the end nodes register their own ID-to-locator mappings with their home domain’s IMA (Fig. 11(0)). Through the procedure shown in Figs. 11(1)–(5), Node_X obtains the identifier of Node_Y. Next, Node_X obtains the locator of Node_Y (Fig. 11(6)) and then Node_X sends a packet to Node_Y (Fig. 11(7)). When Node_Y returns a reply packet to Node_X, the mapping between the identifier and the locator of Node_X is validated through the procedure shown in Figs. 11(8)–(13). Finally, Node_Y sends a packet to Node_X (Fig. 11(14)).

4.6 Details and Current Status of ZNP

See the paper [14] for other signaling examples. The paper also designs Z Control Message Protocol (ZCMP) and the neighbor discovery protocol for ZNP called Z Neighbor Discovery Protocol (ZNDP). ZCMP manipulates the mapping systems and ZNDP resolves the mapping between the ZNP Locator and link layer address (Locator-to-L2addr mapping). ZNP and ZNDP have been implemented in the liunux-2.6.18 kernel space. ZCMP has been implemented in the Linux-2.6.18 user space.

Some performance evaluations are shown below. The transmission, reception and forwarding times of a UDP/ZNP packet are 12.65 µsec, 5.36 µsec, and 7.96 µsec, respectively, where the total size of the packet is 104 bytes; Ethernet header (14 bytes) + ZNP header (72 bytes) + UDP header (8 bytes) + payload (10 bytes). In case of UDP/IPv6 packet, the transmission, reception, and forwarding times are 12.29 µsec, 18.91 µsec, and 8.81 µsec, respectively. The reason why the reception and forwarding times of ZNP are faster than those of IPv6 is that IPv6 has some functions, e.g., multicast handling, that the current implementation of ZNP does not have.

The ID resolving time (T_{id}) is represented as T_{id} = (T_{proc} + RTT) × (n + 1), where T_{proc} is the query processing time of the NMA or the IMA, RTT is the round trip time, and n is the number of hierarchy levels of the node name. The locator resolving time (T_{loc}) is represented as T_{loc} = (T_{proc} + RTT) × (n + 2). If the requesting node has the
cache, \(T_{id} = T_{loc} = 0\). As a result of our measurement, \(T_{proc}\) is approximately 38.65 \(\mu\)sec. \(T_{proc}\) is negligible in comparison with \(RTT\), which is usually in msec order.

5. Adaptation to Dynamically Changing Network Conditions by Inter-Node Cross-Layer Cooperation

5.1 Effectiveness of Inter-Node Cross-Layer Cooperation

The following two examples show effectiveness of internode cross-layer cooperation. The first example shows cooperation between the L2 of a router and the L4 of an end node. In Fig. 12, two nodes N1 and N2 are communicating with each other through an SCTP association. An SCTP association can have multiple paths, one of which is used as the primary path and others are reserved as secondary paths. When the primary path fails, one of the secondary paths is engaged as the primary. This procedure is called failover. In the example, the SCTP association has two paths and the upper path is used as the primary. Suppose that link failure-(1) occurs. SCTP in the end node N1 can detect this failure through cooperation between the L2 and the L4 in N1. Then, N1 can start the failover procedure. Suppose that link failure-(2) occurs. Both end nodes, however, cannot detect this link failure through intra-node cross-layer cooperation. If the L4 in N1 and L2/L3 in R1 cooperate, the L4 in N1 can detect link failure-(2) immediately and start the failover procedure.

The second example shows cooperation between the L2 and the L3 of a mobile router (MR) in a mobile network and the L4 of a mobile node in the mobile network. In Fig. 13, a mobile network is connected to the global Internet through a WiFi link. A mobile node (MN) in the mobile network is communicating with a correspondent node (CN) in the global Internet. The mobile network is going to handover from a WiFi access router to a 3G access router. As a result, the bandwidth of the wireless link is reduced from 54 Mbps to 2 Mbps. Without an inter-node cross-layer cooperation mechanism, the MN cannot detect bandwidth reduction due to handover and keeps the transmission rate (the congestion window size in case of TCP) after handover. This causes heavy congestion at the MR. If the L4 in the MN and L2/L3 in the MR cooperate, the L4 in the MN can detect that a handover to a lower access point is about to occur and adjust the congestion window size so that congestion at the MR is avoided.

5.2 Requirements to Cross-Layer Cooperation Mechanisms

In the two examples described in Sect. 5.1, control information is transferred from a layer in a node to a layer in another node and the two layers are not the same layer. In the two examples, control information is exchanged between the L4 in an end node and the L3 or the L2 in a router. There are several protocols in the L4, the L3, and the L2, and there are a lot of device types in the L2. To enable such interactions, control information must be transformed from protocol- or device-specific information to protocol- or device-independent information. In addition to information abstraction, each layer must define primitives as a clearly-defined API, which makes inter-node cross-layer cooperation easier. In the second example in Sect. 5.1, the two nodes that exchange control information are basically connected to the same subnet while in the first example, the two nodes that exchange control information might be located a lot of hops away. Therefore, an inter-node cross-layer cooperation mechanism must be available in a wide area network.

As a result of the discussion above, we defined the following five requirements to inter-node cross-layer cooperation architectures: (1) cooperation between a layer in a node and a layer (not necessary the same layer) in another node, (2) abstraction of control information, (3) definition of primitives in each layer, (4) availability in wide area networks, and (5) implementation. The fifth requirement means that an inter-node cross-layer cooperation architecture can be implemented in real systems.

A lot of cross-layer cooperation architectures have been proposed [36]–[45]. Some of them focus on mobility in wireless communication environment while some of them focus on multimedia communication. Most cross-layer cooperation architectures in the literature consider cooperation of layers within a node (intra-node). In addition to intra-node cross-layer cooperation, inter-node cross-layer cooperation should be employed to efficiently adapt to the dynamically changing network conditions. None of the proposals described above satisfy the five requirements. CLINEX was designed so that it satisfy the requirement.

5.3 CLINEX Architecture

Figure 14 shows the CLINEX architecture. The OSI ref-
CLINEX employs the Abstract Entity (AE), the Inter-Layer System (ILS), and the Inter-Node System (INS) in each node. An AE is attached to each PE. The ILS vertically penetrates all layers like a skewer. An AE abstracts protocol-specific control information to protocol-independent control information (the requirement-(2)). Abstracted control information is exchanged between layers that might not be adjacent through the ILS. The INS exchanges abstracted control information between nodes (the requirement-(1)). The L3 control plane is responsible for abstracted control information exchange between nodes in wide area networks (the requirement-(4)). In addition, the KB (Knowledge Base) is installed in representative nodes of administrative areas such as the areas of OSPF (Open Shortest Path First), one of the standard intra-AS routing protocols. The KB is a database that stores information of the network such as the topology of the coverage area and available bandwidth of each link in the coverage area.

5.4 Message and Service Types

CLINEX defines four classes of messages: the Request (req), the Confirm (conf), the Indication (ind), and the Response (res). The Request is used when an AE in a layer (e.g., the L3) requests some action of an AE in the target layer (e.g., the L2). The Confirm is the reply to the Request. The Indication notifies the requesting AE in a layer (e.g., the L3) of occurrence of an asynchronous event in the target AE in another layer (e.g., the L2). The Response is the reply to the Indication.

CLINEX defines three service types as shown in Fig. 15. In the Information service, a requesting AE in a layer (e.g., the L3) sends the Request to the target AE in another layer (e.g., the L2). The target AE replies with the Confirm to the requesting AE. In the Event Service, a requesting AE in a layer registers a request with the target AE in another layer through the Request. The target AE replies with the Confirm as an acknowledgment of the Request. When the requested event occurs in the target AE, it notifies the requesting AE of the event through the Indication. The requesting AE would reply with the Response to the target AE. In the Command Service, the requesting AE in a layer sends the Request to the target AE to execute some action. The target AE replies with the Confirm as an acknowledgment of the Request.

5.5 L2 and L3 Primitives

To satisfy the requirement-(3) (definition of primitives), CLINEX defines primitives in the L2 and the L3 as the first step of its design.

5.5.1 L2 Primitives

It can be assumed that indispensable control information in the L2 is regarding physical interfaces and L2 links. Therefore, it would be sufficient to provide an AE in other layers with the following information: what kinds of physical interfaces are installed; what the status of an interface is; what node is connected through an interface via a wired link; what node could be connected through an interface via a wireless link; and what the status of the L2 link is. Thus, CLINEX defines the ten L2 primitives as shown in Table 1. There are four Information Service (IS) primitives, four Event Service (ES) primitives, and two Command Service (CS) primitives.

5.5.2 L3 Primitives

It can be assumed that indispensable control information in the L3 is regarding interfaces visible in the L3 and paths between a pair of nodes. Therefore, it would be sufficient to provide an AE in other layers with the following information: what kinds of interfaces are visible in the L3; what the

---

**Table 1 L2 primitives in CLINEX.**

<table>
<thead>
<tr>
<th>type</th>
<th>primitive</th>
<th>function</th>
</tr>
</thead>
<tbody>
<tr>
<td>IS</td>
<td>L2-IfList</td>
<td>query of if/ list in L2</td>
</tr>
<tr>
<td>IS</td>
<td>L2-IfStat</td>
<td>query of if/ status in L2</td>
</tr>
<tr>
<td>IS</td>
<td>L2-PoAList</td>
<td>query of PoA list</td>
</tr>
<tr>
<td>IS</td>
<td>L2-LinkStat</td>
<td>query of link status</td>
</tr>
<tr>
<td>ES</td>
<td>L2-IfListChanged</td>
<td>notification that if/ list changed</td>
</tr>
<tr>
<td>ES</td>
<td>L2-IfStatChanged</td>
<td>notification that if/ status changed</td>
</tr>
<tr>
<td>ES</td>
<td>L2-PoAListChanged</td>
<td>notification that PoA list changed</td>
</tr>
<tr>
<td>ES</td>
<td>L2-LinkStatChanged</td>
<td>notification that link status changed</td>
</tr>
<tr>
<td>CS</td>
<td>L2-LinkConnect</td>
<td>command to connect to specified PoA</td>
</tr>
<tr>
<td>CS</td>
<td>L2-LinkDisconnect</td>
<td>command to disconnect from a PoA</td>
</tr>
</tbody>
</table>

---

![Fig. 14 CLINEX architecture.](image)

![Fig. 15 CLINEX interaction model.](image)
status of an interface is; what the status of a path to another node is; and what the status of the routing information is.

Thus, CLINEX defines the ten L3 primitives as shown in Table 2. There are four Information Service (IS) primitives, four Event Service (ES) primitives, and two Command Service (CS) primitives. IN-IS means inter-node IS and IN-ES means inter-node ES.

### Table 2 L3 primitives in CLINEX.

<table>
<thead>
<tr>
<th>type</th>
<th>primitive</th>
<th>function</th>
</tr>
</thead>
<tbody>
<tr>
<td>IS</td>
<td>L3-IS-IfList</td>
<td>query of i/f list in L3</td>
</tr>
<tr>
<td>IS</td>
<td>L3-IS-IfStat</td>
<td>query of i/f status in L3</td>
</tr>
<tr>
<td>IN-IS</td>
<td>L3-PathStat</td>
<td>query of path status</td>
</tr>
<tr>
<td>IS</td>
<td>L3-RoutInfo</td>
<td>query of route info.</td>
</tr>
<tr>
<td>ES</td>
<td>L3-IS-IfListChanged</td>
<td>notification of i/f list change</td>
</tr>
<tr>
<td>ES</td>
<td>L3-IS-IfStatChanged</td>
<td>notification of i/f status change</td>
</tr>
<tr>
<td>IN-ES</td>
<td>L3-PathStatChanged</td>
<td>notification of path status change</td>
</tr>
<tr>
<td>ES</td>
<td>L3-RoutChanged</td>
<td>notification of route info. change</td>
</tr>
<tr>
<td>CS</td>
<td>L3-RouteAdd</td>
<td>command to add routing info.</td>
</tr>
<tr>
<td>CS</td>
<td>L3-RouteDelete</td>
<td>command to remove routing info.</td>
</tr>
</tbody>
</table>

#### 5.6 L3-PathStatChanged Processing

Due to space limitation, this paper shows an example of the CLINEX procedures. Figure 16 shows the CLINEX procedure in the SCTP failover procedure shown in Fig. 12. Note that the Confirm messages (e.g., L3-PathStatChanged.conf) as the acknowledgement of the Request messages (e.g., L3-PathStatChanged.req) are omitted in the figure. First, Node-1 and Node-2 start SCTP communication via the primary path (Fig. 16(0)). Suppose that SCTP (L4) in Node-1 wants to know status change on the primary path. SCTP in Node-1 sends the L3-PathStatChanged.req message containing the information about both end nodes to KB node (Fig. 16(1)). Since the KB node knows that R1, R2, and R3 are on the primary path, it sends the L3-PathStatChanged.req message to those nodes (Fig. 16(2)). Note that a single L3-PathStatChanged.req message is sent by the KB node. The message is copied in R2 and forwarded to R1 and R3. In R1, R2, and R3, upon receiving the L3-PathStatChanged.req message, the L3 sends the L2-LinkStatChanged.req message to the L2 to detect link failure. Suppose that link failure occurs on the link between R2 and R3 (Fig. 16(3)). The L2 of R2 detects this link failure and it sends the L2-LinkStatChanged.ind message to the L3 of R2. Upon receiving the message, since the L3 knows that the status of the path specified by the L3-PathStatChanged.req changed, it sends the L3-PathStatChanged.ind message to the KB node (Fig. 16(4)). Upon receiving the L3-PathStatChanged.ind message, the KB node forwards it to Node-1 (Fig. 16(5)). As a result, Node-1 performs failover to the secondary path (Fig. 16(6)).

### 5.7 Details and Current Status of CLINEX

See the paper [15] for the detailed procedure of CLINEX. The main part of CLINEX has been implemented in the kernel space and the user space of Ubuntu 12.04.1 LTS (linux-3.2.0). The ILS is implemented in the kernel space and the INS is implemented in the user space. Some AEs are implemented in both the kernel space and the user space.

One of performance evaluations is shown below. Suppose that an application process on a node wants to know occurrence of congestion on its communication path and a router on the communication path detects congestion. On the router, an L3-AE in the kernel space detects congestion and it notifies an L3-AE in the user space of the event (38 μsec). The L3-AE in the user space generates an L3-PathStatChanged.ind message (16 μsec) and sends the message to the ILS in the kernel space (7 μsec). The ILS judges that the message is an inter-node message (5 μsec) and it sends the message to the INS in the user space (45 μsec). The INS in the user space sends the message to the node (23 μsec + delay from the router to the node). On the node, the INS in the user space receives the message and knows that the message is an L3-PathStatChanged.ind message (25 μsec) and sends it to the ILS in the kernel space (9 μsec). The ILS in the kernel space judges that this message should be sent to L6 (the application layer)-AE (7 μsec) and it sends the message to the L6-AE in the user space (25 μsec). The L6-AE receives the message (21 μsec) and notifies the application process that congestion occurred on the router (54 μsec). Thus, it takes 275 + delay μsec from the time when the L3-AE in the router detects congestion to the time when the application process receives the notification, where delay is the communication delay from the router to the node.

### 6. Conclusion

This paper introduced a new network architecture called ZNA for New Generation Network. First, this paper discussed the layered model, cross-layer cooperation, information centric networking, and ID/Locator split. As a result, ZNA adopts a six-layer model with inter-node cross-layer cooperation and ID/Locator split. Among the features of ZNA, this paper introduced three of them: (1) sophisticated services in the session layer, (2) support of mobility, multihoming, and heterogeneity of network layer protocols, and (3) inter-node cross-layer cooperation.

As the sophisticated services, ZNA defined three kinds of communication paths in the session layer: the bundled-path, the spatially-spliced-path, and the temporally-spliced path. The session layer API was defined and an example of a bundled path establishment procedure was shown. As the
network layer protocol for NwGN, four requirements were defined. ZNP was designed so that the four requirements were satisfied. ZNP adopted ID/Locator split approach to support mobility, multi-homing, and heterogeneity of network layer protocols. The mapping systems of ZNP were also introduced and an example of ID and locator resolution was shown. As the cross-layer cooperation for NwGN, five requirements were defined. CLINEX was designed so that the five requirements were satisfied. CLINEX realizes exchange of control information between layers not only in the same node but also in different nodes to adapt to the dynamically changing network conditions. An example of CLINEX signaling was shown, in which the L4 in an end node cooperates with the L2 and L3 on a router so that the L4 in the end node can detect L2 failure in the router.

Currently, some parts of ZNA have been implemented. We are revising the design and the protocols of ZNA reflecting the findings through the implementation. There are protocols and functions that have not yet been implemented. We continue improvement and implementation of ZNA. We would like to build a test network based on ZNA using a network virtualization technique in the near future.

References


**Fumio Teraoka** is a professor of Faculty of Science and Technology, Keio University. He received a master degree in electrical engineering and a Ph.D. in computer science from Keio University in 1984 and 1993, respectively. He joined Canon Inc. in 1984 and then moved to Sony Computer Science Labs., Inc. (Sony CSL) in 1988. Since April 2001, he is a professor of Faculty of Science and Technology, Keio University. He received the Takahashi Award of JSSST (Japan Society for Software Science and Technology) and the Motoooka Award in 1991 and 1993, respectively. He also received the Best Paper Award in 2000 from IPSJ (Information Processing Society Japan). His research interest covers computer network, operating system, and distributed system. He contributed to the activity of the Mobile working group of IETF by developing Virtual IP (VIP). He was a board member of the WIDE Project from 1991 to 2010. He was a board member of IPSJ from 2000 to 2002. He was a board member of JSSST from 2005 to 2009. He is a member of ACM, IEEE, JSSST, IPSJ, and IEICE.

**Sho Kanemaru** received a master degree in School of Science for Open and Environmental Systems, Graduate School of Science and Technology, Keio University, Japan in 2013. He is interested in new generation network technologies, especially, ID/Locator split approach and cross-layer cooperation. Currently, he works at NTT Network Service Systems Laboratories.

**Kazuma Yonemura** received a master degree in School of Science for Open and Environmental Systems, Graduate School of Science and Technology, Keio University, Japan in 2014. He is interested in new generation network technologies, especially, sophisticated communication services in the session layer. Currently, he works at DWANGO Co., Ltd.

**Motoki Ide** received a master degree in School of Science for Open and Environmental Systems, Graduate School of Science and Technology, Keio University, Japan in 2013. He is interested in new generation network technologies, especially, ID/Locator split approach and cross-layer cooperation. Currently, he works at Nomura Research Institute.

**Shinji Kawaguchi** is a master course student in School of Science for Open and Environmental Systems, Graduate School of Science and Technology, Keio University, Japan. He is interested in cross-layer cooperation technologies.

**Kunitake Kaneko** is an assistant professor of Faculty of Science and Technology, Keio University. He received his M.S. from Graduate School of Engineering, the University of Tokyo, and his Ph.D. from the Graduate School of Information Science and Technology. In 2006, he joined Research Institute for Digital Media and Technology (DMC) at Keio University. His research interest is in new generation network architecture from network applications’ perspective. He is a member of IEEE, SMPTE, IPSJ, and IEICE.