Network Forensics in a Clean-Slate Internet Architecture

Tinus Strauss
Department of Computer Science
University of Pretoria, South Africa
Email: tstrauss@cs.up.ac.za

Martin S. Olivier
Department of Computer Science
University of Pretoria, South Africa
Email: www.mo.co.za

***Abstract*—This paper reflects on the network forensic implication of a specific clean-slate future internetwork architecture. The paper first provides an overview of the architecture and how it compares to the well-established TCP/IP model. The architecture's network forensic features are then considered.**

The architecture's approach to naming and addressing fundamentally differs from the approach used in the current Internet. Great care is taken to distinguish between names and addresses. Names are used to identify entities and generally have a large scope. Addresses, however, are used to locate entities within a limited scope and are consequently not necessarily globally significant. These properties in particular create additional challenges when capturing and analysing network traffic as evidence.

The paper shows that the architecture is well-suited for a distributed systems approach to forensics and that the network architecture increases the potential sources of reliable evidence.

I. Introduction

The Internet started as an experimental network many years ago. The initial designers did not anticipate the phenomenal success of the Internet as we know it today. It is only natural that the original designers have not considered the present operational demands in their design decisions. As a consequence the Internet suffers from a number of problems [1]. Among these are address depletion, core routing table size and providing support for multihoming, multicast, and mobility.

To ensure the future success of the Internet, these problems need to be addressed. Some evolutionary initiatives include Internet Protocol version 6 (IPv6), Global, Site, and End-system address elements (GSE) [2], and Locator Identifier Separation Protocol (LISP) [3]. However; there are those who argue that a clean-slate research approach is advantageous since one is not constrained by the limitations of the current network architecture [4]. This has caused several longer term research efforts [5]–[7] to form in which researchers are investigating future Internet architectures.

One such proposed architecture is called the Recursive Internet Architecture (RINA) and is detailed by Day [8]. This architecture is derived from the notion that computer networking is essentially a form of interprocess communication (IPC). The security aspects of the architecture have been assessed by Boddapati et al. [9]. In the present work the forensic aspects are considered. It should be noted that specific protocols are not detailed or analysed. The work considers the elements of the architecture, their functions, and their interactions.

The paper is structured as follows. First RINA is described in section II. Here the elements of the architecture and their interactions are described. The section also explains how the architecture supports IPC. After the architecture has been described, the paper considers the forensic properties of the architecture in section III. The paper concludes in section IV.

II. Recursive Internet Architecture

As mentioned earlier, the recursive internet architecture (RINA) [8], [10] uses IPC as its guiding principle. It makes a break from the traditional OSI or TCP/IP layered models and approaches the networking problem from a fundamental basis—two processes wishing to exchange data.

In the layered architectures, a layer provides a service to the layer directly above it. Each layer encapsulates certain functionality. For example, the data link layer provides reliability over the physical medium, the network layer provides routing and relaying, and the transport layer provides end-to-end error- and flow-control. In the IPC approach the transport task and the network task together provides an IPC service to an application process. A given IPC facility provides a service over a given scope. It is possible to stack IPC services to that an IPC service utilises the service of an IPC facility at a lower level. Thus, in this model, the IPC facilities repeat. They are functionally the same but they are applied to different scopes or domains, so the policies governing their behaviour might differ. In the traditional approaches the layers are used to isolate different functions. However; in the IPC approach; the layering is used to support different ranges of the resource allocation problem.

First will be looked at the special case of two processes on two different systems communicating via a direct physical connection. The scenario subsequently will be expanded to the general case in which one or more relaying agents are required to establish communication.

A. Two directly connected processes

Consider first the problem of two application processes wishing to communicate. Each process is located in a distinct processing system and the systems are connected via a direct communication channel. Recall that the architecture provides the application processes with a distributed IPC facility.

The IPC facility is responsible for locating the destination application process using a name; allocating resources to establish the communication channel; and providing a port number to the application. This port-id is like a file descriptor which can be used to send and receive data.

Figure 1 illustrates the scenario. The application process uses the service of the distributed IPC facility (DIF). The port-id is used as a service access point into the facility. It should also be pointed out that it is the application protocol component of the application which communicates with a peer application protocol. An application could have more that one application protocol.

Within a given system the IPC service is provided by an IPC process. The IPC process is represented by the large circle in the figure. This process consists of a number of subtasks and protocol machines. The figure depicts a number of these components.

- An IPC manager orchestrates and manages the other tasks.
- A routing information exchange protocol (RIEP) to do routing and connection establishment. It uses the resource information base (RIB).
- An IPC allocation protocol (IAP) which is used to find the other application process and to determine whether communication should be allowed. If allowed, it enacts resource allocation to support the communication.
- A data transfer control protocol (DTCP) controls data transfer, error, and flow control. Delta-t [11] is suggested by the designers.
- A routing and multiplexing task (RMT) which is responsible for scheduling access to the underlying physical medium.

In this simple two system case, only one DIF is needed. In the subsequent section, the case is extended to include relaying.

B. Relaying introduced

Here the problem is generalised to the scenario where communication needs to occur between hosts in arbitrary points in a network. Since cost typically prevents fully connected networks, this implies that some form of relaying is required.

Figure 2 gives a simplified representation containing two hosts and a relaying station. The relay is analogous to a router in the TCP/IP world. Note that the figure shows two DIFs. The end-to-end communication between the applications is supported by the upper layer DIF. The upper DIF thus provides an end-to-end transport service, but also includes routing between nodes (IPC process) within the DIF. The RMT task in the upper layer multiplexes data unit to the lower level DIF.

Note that the IPC process in the relay is different from the IPC process in the hosts in that it is a dedicated relaying application. It has multiple instances of RMT since a routing decision will determine which lower layer DIF services should be used.

The lower layers DIFs are media dependent DIFs and correspond to the traditional data-link layer. The DTCP here provides a error and flow control protocol over the physical medium. The policy should be such that it suits the medium. The RMT task multiplexes data onto the medium. Depending on the medium, the RMT could also provide routing functions. Consider for example, spanning trees in Ethernet.

The layering approach in RINA is different from the layers in OSI. Here the layers provide IPC for a given scope. In this simple configuration the one scope is end-to-end between the application protocols. The other scope is point-to-point between the "wires". Note that the DIFs are essentially the same; they only differ in terms of scope and consequently the policies used by the mechanisms. The model can be expanded where more overlays are required. Consider network-based virtual private networks as an example. In general, an (N)-DIF uses the services of an (N–1)-DIF and provides a service to an (N+1)-DIF. The number of layers may vary depending on the requirements in the network. The lowest layer DIF is usually an IPC process geared to the particular physical medium. The case in Figure 2 is a fairly standard case, relating closely to the standard TCP/IP model.

C. Operation

All communication in RINA goes through the following phases: enrollment, allocation, and data transfer. That is, an IPC process must first be granted access to a DIF and then, when communication is necessary, resources are allocated to be able to serve the communication requirements and only when this has successfully been accomplished is data transferred. The architecture does not dictate the policies to be used for these functions, so it is possible that a null policy is applied which would entail that the function is not required.

1) Enrollment: Enrollment entails "becoming part of the network". There are two variations—joining an existing DIF and creating a new DIF. The latter is a simple process. A network management process creates a new IPC process with appropriate permissions and points it to the appropriate (N–1)-DIF. It should also be provided with the means to recognize other IPC process to allow to join the DIF. It can wait for other IPC processes to join or it can contact others.

If a DIF has been created, other IPC process may need to join the DIF. Assume IPC process b wants to become a member of (N)-DIF A. Assume further that it is currently in (N)-DIF B consisting of only itself. A precondition for joining is that DIF A and DIF B need to be connected via a common (N–1)-DIF (or physical medium) and b needs to know the name of DIF A or the name of a member of DIF A, say a. b uses the (N–1)-DIF to establish an IPC connection to a using a's name.

It is important to note the difference between name and address. Applications outside of a DIF, using the services of a DIF, access the services of the DIF using names and not addresses. Names are global, but addresses are private. b has no way to know any of the addresses of elements of DIF A.

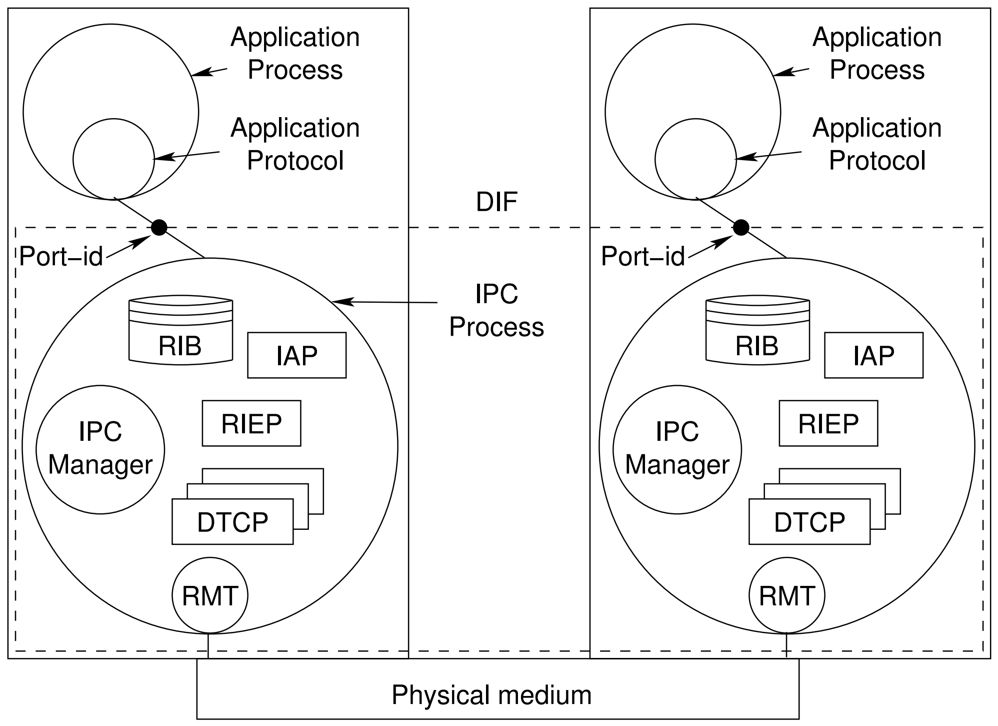

Fig. 1. IPC with two directly connected systems.

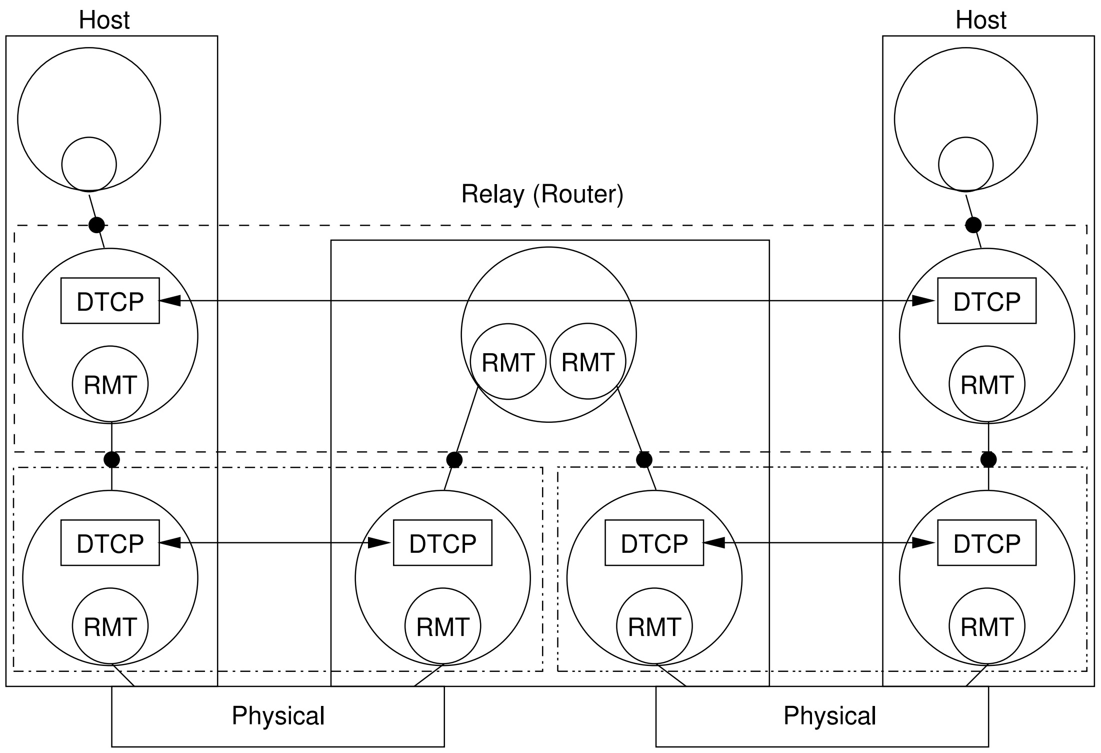

Fig. 2. General model with a relay node.

Once the link has been established, a authenticates b and determines whether b may join the DIF A. If policy allows b to join, b is assigned an (N)-address which it can use to identify it to other members of the DIF A. Other parameters associated with DIF A may also be communicated to b. b is now a member of DIF A. It will also go through this process with other directly connected neighbours. Since the (N–1)-DIF may have a smaller scope than DIF A, there may be members of DIF A which b cannot reach via an (N–1)-DIF. This is where relaying nodes come in and where the RIEP is required.

2) Resource allocation and data transfer: Once the enrollment phase has completed, sufficient shared state exists among the IPC management processes to allow the DIF to provide IPC services to applications (or to act as an relay, in the case of a relay).

When two application need to communicate, they use the IPC service of a DIF. The first step is to allocate resources to support the communication. This entails finding the remote application and creating bindings between the application and the relevant components of the IPC process.

Assume that application process A wishes to communicate with application process B on some remote system. A needs to establish an IPC connection with B. A uses the services of a DIF represented by the IPC process with name a. A uses library calls to interact with a. It creates an `allocate` request which includes A's name, a's address, a local port-id, B's name, and connection policy parameters. If the request is valid then, since B is not local, a's local management task needs to locate B. This is done using an IPC allocation protocol (IAP). According the defined search rules, the IAP request will be forwarded to other IPC process until b is found[1]. b now determines whether it will allow communication and whether it can honour the connection parameters. If it does, it will assign a local port number and send an IAP response back to a. Once a receives the response, the `allocate` request returns and provides A with the parameters required for the communication. Also a allocates a DTCP instance and binds it to the local port-id provided to A. The application A may now start sending application protocol data units (PDUs) to its peer B.

The application PDUs are received by the IPC facility as service data units (SDUs). The SDU is then delimited and inserted as data into a PDU for the DTCP corresponding to the port-id used by the application. These PDUs are then processed by the RMT for transmission. The RMT may multiplex multiple PDUs into a single (N–1)-SDU which will be serviced by the (N–1)-DIF.

When A has finished its communication with B, it may perform a `deallocate` request. This will result in the release of the resources allocated to this IPC instance.

III. Forensics in RINA

Network forensics is defined [12] as "the use of scientifically proven techniques to collect, fuse, identify, examine, correlate, analyze, and document digital evidence from multiple, actively processing and transmitting digital sources for the purpose of uncovering facts related to the planned intent, or measured success of unauthorized activities meant to disrupt, corrupt, and or compromise system components as well as providing information to assist in response to or recovery from these activities."

Since attribution is an important part of forensics, consider in summary the identifiers used in RINA. Building on work by Saltzer [13] the identifiers used in the architecture are as follows. Three identifiers are externally visible, one is internal to the processing system and two are internal to a DIF. The three external identifiers relate to the applications: Distributed application names that identify a set of application processes cooperating to perform a function; application process names to identify application processes; and application protocol machine names, used to distinguish application protocol machines within an application process. It should be clear that the application related entities are defined by names and not addresses.

The identifier internal to the processing system is the port-id. Recall that this is essentially an operating system facility to allow the application process and the IPC process to communicate. The two identifiers internal to the DIF are (N)-addresses assigned to the IPC processes and the connection-id used in the DTCP protocol to distinguish connections. The connection-id is created by the pair of port-ids in the two systems partaking in the particular instance of the DTCP. Note that addresses are not exposed outside the DIF.

Many forensic processes and frameworks have been defined and the following steps are representative [14].

a) Preparation: Forensics is only applicable where relevant forensic tools are in place. In addition to the regular forensic tools, the IPC model allows for additional sources of evidence. For example, enrollment must be performed explicitly. The processing system could include a forensic agent which has sufficient permissions log the state of the IPC tasks.

b) Detection: Detecting anomalous behaviour is usually the function of an system external to the normal operation of the network. An intrusion detection system is one example. In RINA, internal functions could also be used to detect abnormal behaviour. Good candidates include the enrollment process and the resource allocation process. Recall that these all depend on configured policies. Requests violating policies could trigger alarms which could then start a process to record detailed data.

c) Incident response: Depending on the nature of the incident, responses may vary. One approach could be to remove a suspicious IPC process from a DIF. Another could be to adjust the resource allocation policies dynamically to limit the effect of the suspicious behaviour.

Previous work [15] has illustrated how one could use an IP-based scheme to isolate logically certain traffic deemed to be of forensic interest. To achieve a similar logical isolation scheme in RINA is fairly straightforward since the required elements are already present in the architecture.

[1]One could think of IAP as an enhanced DNS.

A special forensic DIF could be inserted into the appropriate position in the environment. Targeted communications could then be directed to use the IPC services of this DIF. The policies in the IPC processes here could be such that traffic is logged in more detail and perhaps other characteristics could be tuned. For example, the selection of the lower layer DIF to utilise for transmitting the data. It could be configured that the traffic then follows a different path through the network which allows the operator to record and preserve the traffic.

d) Collection: The forensic tools (sensors) should store data. In RINA it is important to not only store the transmitted traffic, but to also store the bindings between processes. Consider for example the DTCP mention earlier. The header of the protocol only transmits the connection-id, which is only meaningful in the context of the IPC processes involved in the communication. If the traffic was captured directly from a physical medium, the header information would not be meaningful without knowing the context. The agents on the processing system would need to record the bindings between port-ids, addresses and application names.

e) Preservation: Clearly the collected data would need to be preserved. The logical isolation scheme mentioned above could also aid in this.

f) Examination, analysis, and investigation: Fusing of data from different sources on which to perform analysis is especially important in RINA. As discussed earlier, the network traffic itself does not provide enough context to interpret the header information. The logs from the IPC processes are required to provide the mappings to applications. This then makes it hard to perform real-time analysis on network traffic. This is an artifact of the encapsulation provided by the DIFs.

Since addresses are not known outside the IPC process in a DIF, it makes spoofing of addresses much harder and consequently attribution may be done more reliably.

IV. Conclusion

This paper considered the RINA clean-slate architectural model from a forensics perspective. Since communication always follows the phases of enrollment, allocation, and data transfer; it allows for forensic abilities to be included inside the architecture. The need for external probes and tools are thus reduced.

Logical traffic isolation is supported quite readily due to the flexible nature of the recursive layering structure.

It was found that RINA provides more sources for digital evidence. Consequently there is a higher load on fusing the different sources to be able to interpret protocol control data. However; this results in more reliable evidence.

Future work should *inter alia* investigate the behaviour of specific protocols and identify and characterise attack profiles.

References

[1] D. Meyer, L. Zhang, and K. Fall, "Report from the iab workshop on routing and addressing," RFC 4984, September 2007.

[2] M. O'Dell, "GSE – an alternate addressing architecture for IPv6," Internet Draft, February 1997, http://tools.ietf.org/html/draft-ietf-ipngwg-gseaddr-00.

[3] D. Farinacci, V. Fuller, D. Meyer, and D. Lewis, "Locator/ID separation protocol (LISP)," draft-ietf-lisp-09, October 2010, work in progress.

[4] J. Rexford and C. Dovrolis, "Future internet architecture: clean-slate versus evolutionary research," *Commun. ACM*, vol. 53, no. 9, pp. 36–40, 2010.

[5] "European future internet portal," http://www.future-internet.eu/.

[6] "Future internet design," http://find.isi.edu/.

[7] "Stanford university clean-slate program," http://cleanslate.stanford.edu/.

[8] J. Day, *Patterns in Network Architecture: A return to fundamentals*. Prentice-Hall, 2008.

[9] G. Boddapati, J. Day, I. Matta, and L. Chitkushev, "Assessing the security of a clean-slate internet architecture," Computer Science, Boston University, Tech. Rep. BUCS-TR-2009-021, 2009.

[10] J. Day, I. Matta, and K. Mattar, "Networking is ipc: a guiding principle to a better internet," *Proceedings of the 2008 ACM CoNEXT Conference*, pp. 1–6, 2008.

[11] R. Watson, "Timer-based mechanisms in reliable transport protocol connection management," *Computer Networks*, vol. 5, no. 1, pp. 47–56, 1981.

[12] G. Palmer, "A road map for digital forensic research," DFRWS, Tech. Rep. DTR-T001-01, 2001, report From the First Digital Forensic Research Workshop.

[13] J. Saltzer, "On the naming and binding of network destinations," RFC 1498, August 1993.

[14] E. S. Pilli, R. Joshi, and R. Niyogi, "A framework for network forensic analysis," in *Information and Communication Technologies*, ser. Communications in Computer and Information Science, V. V. Das and R. Vijaykumar, Eds. Springer Berlin Heidelberg, 2010, vol. 101, pp. 142–147.

[15] T. Strauss, M. Olivier, and D. Kourie, "Differentiated services for logical traffic isolation," in *Advances in Digital Forensics II*, M. Olivier and S. Shenoi, Eds. Springer, 2006, pp. 229–237.