# Strengthening the Security of Cognitive Packet Networks

Ricardo Lent

Intelligent Systems and Networks Group, Electrical and Electronic Engineering, Imperial College, SW7 2BT, London, UK

Georgia Sakellari

School of Architecture, Computing and Engineering, University of East London, E16 2RD, London, UK

George Loukas

School of Computing and Mathematical Sciences, University of Greenwich, SE10 9LS, UK

**Abstract:** Route selection in Cognitive Packet Networks (CPN) occurs continuously for active flows and is driven by the users' choice of a Quality of Service (QoS) goal. Because routing occurs concurrently to packet forwarding, CPN flows are able to better deal with unexpected variations in network status, while still achieving the desired QoS. Random neural networks (RNN) play a key role in CPN routing and are responsible to the next-hop decision making of CPN packets. By using reinforcement learning, RNNs' weights are continuously updated based on expected QoS goals and information that is collected by packets as they travel on the network experiencing the current network conditions. CPN's QoS performance had been extensively investigated for a variety of operating conditions. Its dynamic and self-adaptive properties make them suitable for withstanding availability attacks, such as those caused by worm propagation and denial-of-service attacks. However, security weaknesses related to confidentiality and integrity attacks have not been previously examined. Here, we look at related network security threats and propose mechanisms that could enhance the resilience of CPN to confidentiality, integrity and availability attacks.

**Keywords:** network security; cognitive packet network; network performance; integrity; confidentiality

## 1 Introduction

Reliable networks that provide good service quality are expected to become the norm in all communication aspects, especially as the information transferred between network users gets more complex and confidential, and as malicious users try to deliberately degrade or altogether deny legitimate network service. There is therefore an increased need for network stability and reliability which has led to the growth of autonomic networks that use QoS-driven approaches for greater stability and reliability in communications. The Cognitive Packet Network (CPN) that was introduced by Gelenbe et al. (1999) has been shown to provide good network adaptation properties under varying network conditions and user requirements. Its performance has been investigated extensively and for a variety of performance metrics, but most studies have considered operating conditions where there is no malicious intent or security threats limited to availability attacks through worm propagation or network denial-of-service (Sakellari, 2009).

In CPN, the steady-state of random neural networks running on network routers decides network paths. Random Neural Networks (RNNs) (Gelenbe et al., 2001b; Gelenbe and Fourneau, 1999) are trained continuously by enhancing network flows with packets that monitor performance. These packets store information about the network state as they cross the network from a source to a destination, and get updated at each hop in the path. In general, CPN blindly trusts the information carried by packets, but a network attack could result in some routers falling under the control of a malicious user, and as a consequence packets may be subject to tampering. The problem is aggravated as routing decisions in CPN are distributed. In this paper, we examine potential weaknesses of CPN and the use of RNNs in relation to confidentiality, integrity and availability, and identify possible solutions that could enhance the resilience of CPN to representative security threats.

## 2 Brief overview of the Cognitive Packet Network

The Cognitive Packet Network (CPN) is a routing protocol that uses adaptive techniques based on on-line measurements to provide QoS to its users (Gelenbe et al., 2000, 2001a; Sakellari, 2009). The users themselves can declare individually their QoS goals, such as minimum delay, minimum packet loss, maximum bandwidth, minimum power consumption or a weighted combination of these. CPN has been designed to perform self-improvement in a distributed manner by learning from the experience of the packets in the network and by constantly probing for the current best routes.

More specifically, CPN uses three types of packets; smart packets (SP) for discovery, source routed dumb packets (DP) to carry the payload and acknowledgement (ACK) packets to bring back information that has been discovered by either SPs or DPs. This information is used in each node to train Random Neural Networks (RNNs) (Gelenbe et al., 2001b) and produce routing decisions. At each network node SPs are routed according to the measured experiences of previous packets with the same QoS goals and the same destination. In order to explore all possible routes and account for sudden network changes, each SP might make a random routing decision instead of the one calculated by the RNN, with a small probability (usually $5-10\%$).

The header of the CPN packets has been modified to allow the packets to gather network information according to the specified QoS goal. Therefore, as packets travel the network, they store QoS data (such as timestamps, counters, etc.) in a special data storage area of the packet header known as the cognitive map (CM) (Gellman, 2007). When a packet arrives at its destination, an acknowledgement (ACK) packet is generated which stores the route taken by the original packet, and the measurements it collected during its journey. The ACK will then return along the reverse route. At each hop the ACK visits, it deposits information in a special short-term memory store called Mailbox. When it finally reaches the source, the ACK establishes the route that the DPs will follow.

At each node a specific RNN, that has as many neurons as the possible outgoing links, provides the SP with the routing decision in the form of an output link. This output link corresponds to the most excited neuron, and since the RNN has a unique solution for any set of weights and input variables, this choice is also unique. The learning process used with RNN is reinforcement learning, which uses the observed outcome of a decision to reward or punish the routing decision, so that future decisions are more likely to improve or maintain the desired QoS goal.

## 3 Identifying and Addressing CPN Security Weaknesses

Here we investigate potential weaknesses of CPN falling under the triad of common security aspects: confidentiality, integrity and availability (Dhillon and Backhouse, 2000) and propose possible solutions. Public and shared networks offer greater security challenges than private and dedicated networks. We will make no assumptions about the type of deployment, so both the discussion and results will be applicable to both cases. However, the discussion will be centered only on network weaknesses and will exclude particular issues at endpoints.

### *3.1 Confidentiality*

Confidentiality is breached when information, held or transmitted through the network, is disclosed to unauthorised individuals or systems. Ensuring total confidentiality in a network is normally very difficult given that packets need regular handling by routers that are managed by a third party, and outside the sender and recipient's domains. In that sense, a CPN is as vulnerable to confidentiality attacks as other types of networks. Up to now there has not been any security mechanism in CPN that could guarantee a confidential end-to-end delivery of users' data.

A confidentiality attack could take place either at hijacked routers or links. In the former case, an attacker could have gained access to a network node on the path from the victim's flow and forward a copy of the traffic content to a collection point. In the latter case, packet sniffers could have been installed on a network link to eavesdrop incoming network traffic. An interesting point to note is that the self-adapting path behaviour of CPN could partially prevent a confidential attack given that not all packets for a given communication may need to pass through the same set of links and nodes unlike traditional networks. A path adaptation may deviate packets away from the hijacked routers and links. However, there is no way to ensure this will happen given that it will be very difficult to detect the packet sniffing process.

The usual solution to ensure network confidentially is to encrypt end-to-end communications, and this approach can be applied also to CPN. In the context of a CPN implementation, at least two alternatives are possible. Data encryption could occur before data is passed to CPN for transmission and after CPN delivers the data (Figure 1, which could be handled either by the application or an intermediate layer above TCP/IP, such as the Transport Layer Security (TLS) protocol. A second possibility is that CPN encrypts and decrypts user data at a per-packet basis at the edge routers or endpoints (Figure 2).

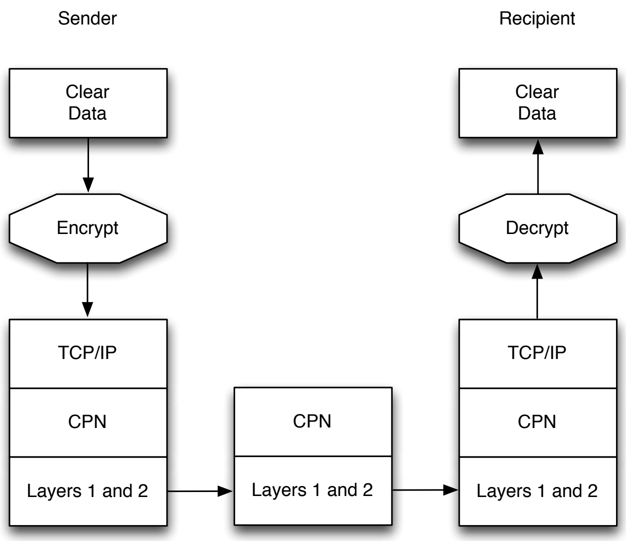

**Figure 1** End-to-end user data encryption and decryption as an external process.

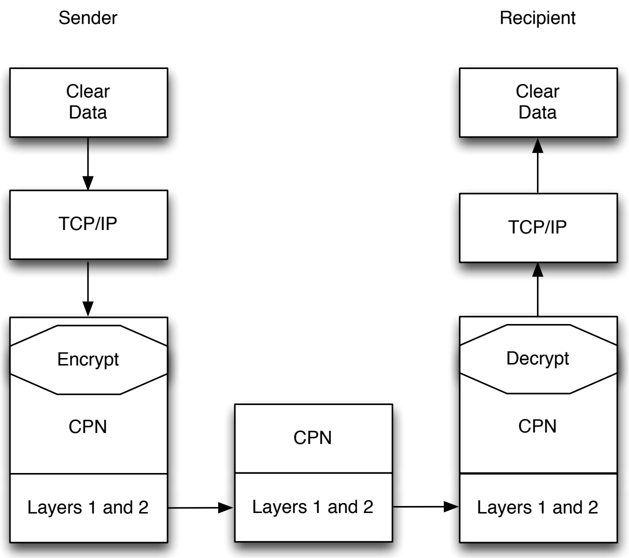

**Figure 2** Per-packet data encryption by CPN.

Figure 3 depicts the transfer time that was measured for files of the indicated length, with or without encryption. The source and destination nodes (which implemented the encryption and decryption process) run with a clock rate of 2.8 and 2.4 GHz respectively. The path connecting the source and destination was unique to ensure obtaining consistent results for all tests, and it consisted of 5 hops. The encryption was realised via secure sockets (so the programming was external to the CPN module). The results indicate that the encryption and decryption process add about one additional second to the total end-to-end transfer time. This additional delay to the file transfer is the penalty caused by the extra computation that was needed.

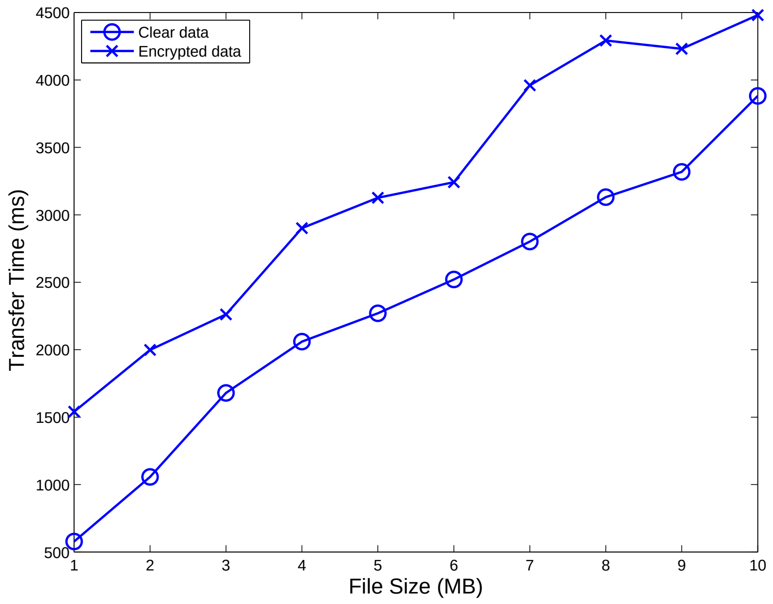

**Figure 3** Measured transfer time over a CPN path of 5 nodes for files of the indicated length with and without encryption.

## *3.2 Integrity*

Integrity refers to the security requirement for information and systems not to be altered without the authorisation of their legitimate owners. As mentioned previously, CPN does not normally encrypt packets, but when done as described earlier, it could also help to (at least partially) address integrity issues given that any alteration of a packet's payload by an attacker will trigger decryption errors at the destination. Unfortunately, not only the users' data could be the target of an attacker, but also the nodes' identity (by changing values in the packet header) and, in the particular case of CPN, network status data and paths (by altering the cognitive map).

Attacks to the CPN header or cognitive map can directly affect the ability of the CPN algorithm and the RNN to effectively respond to network and user behaviour changes, and can also affect the quality of the routing decisions. By changing the packet header or cognitive map, which stores real observations of the QoS that could be achieved by the network, a malicious user could impact the RNN training process, and therefore tweak the routing decisions to cause unexpected behaviour.

Other ways for this to happen is by changing the actual information that is stored in the mailboxes of the nodes, by altering the RNN weights, or even by replacing the RNN-based algorithm with an arbitrary one. Attackers could gain control of one or more nodes in the network and easily change the RNN algorithm behaviour, either directly or indirectly through manipulation of the mailbox entries.

To illustrate the problem, consider a communication flow from nodes $A$ to $B$ as depicted in Figure 4. A normal communication flow will include packets listing in their cognitive map the path $A, C, B$ (step 1). If node $C$ becomes under the control of an attacker, it could rewrite the cognitive map of the packets and forward them to an arbitrary node $D$ (step 2). The purpose could be either to gain access to the packets' contents or to simply affect the quality-of-service of packets (e.g., increase their end-to-end latency). After being intercepted by node $D$, the packets could continue their normal path (step 3). As usual, the destination will create acknowledgements and sent them on the reverse path (step 4) until they reach the attacker's nodes $D$ and $C$ (step 5). Node $C$ could then rewrite the cognitive map back to the original (step 6) to conceal the attack from the source. Several variations of this kind of attack could occur, for example, rather than forwarding packets to node $D$, node $C$ could simply write untrue values in the cognitive map metrics perhaps to indicate better QoS metrics on paths passing through $C$ so that packets become easily available to the attacker. The key CPN weakness in these cases is in having absolute trust in the distributed cognitive map handling.

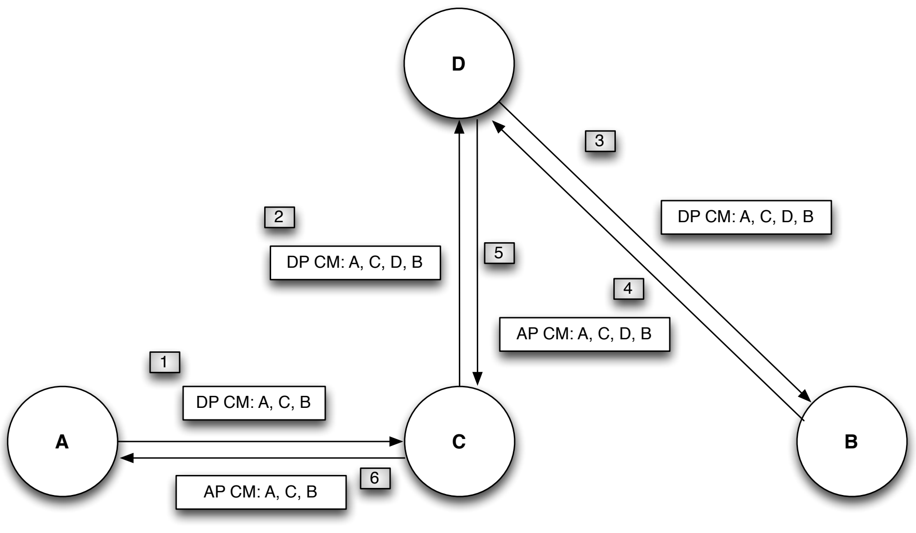

**Figure 4** Example of cognitive map integrity attack.

To prevent such kinds of integrity attacks to the CPN header and cognitive map, we propose introducing digital signatures both at the source (for normal packets) and destination (for acknowledgements). Asymmetric cryptography is a suitable alternative for this task given that it removes the need of secret key distribution. Under this scheme, CPN routers will be required to have both a private and public key.

We suggest two possible implementation approaches. In the first case, we differentiate core routers from edge routers (those connected to the end users), with edge routers signing and verifying signatures for each flow. In a second case, all routers will participate in the signature verification process. The advantage of the former case is speed whereas the second provides greater resilience, as attacked packet could be removed earlier from the network. Another difference between the two is the method of distribution of the public key. In the first case only edges will be required to know the public key of edge routers. In the latter case, all router will have to have access to those keys.

The signature generation and verification process is depicted in Figure 5. The source (end-node or edge router) will generate a message digest of the CPN normal (dumb) packet header and partial cognitive map (including entries that are not expected to change). The message digest will be then encrypted using the source's private key and inserted into the extended CPN header. The extended packet is then sent over the CPN using its regular packet forwarding mechanism. At the receiving end (or intermediate hop, depending on the scheme being used), the signature will be regenerated from the receiving header and cognitive map, and compared with the signature in the packet. If they do not match, something in the header of the cognitive map must have changed on the way, thus indicating the packet has been tampered and should be discarded.

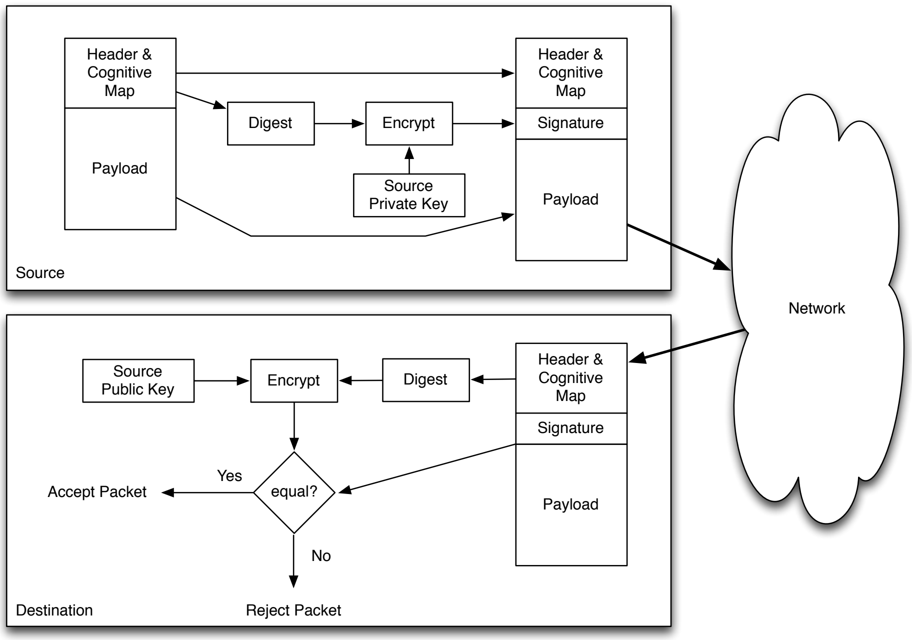

**Figure 5** Digital signature generation and verification for the CPN header and cognitive map.

The signature generation and verification will add additional delay to a normal end-to-end packet transmission. Measurements of these additional delays are depicted in Figure 6 for packets of different length. We have observed that signature generation took around 6.5 ms, signature verification was faster at 3.6 ms, and that the packet length has little impact on the execution time of the processes. The measurements were obtained on a 2.8 GHz machine.

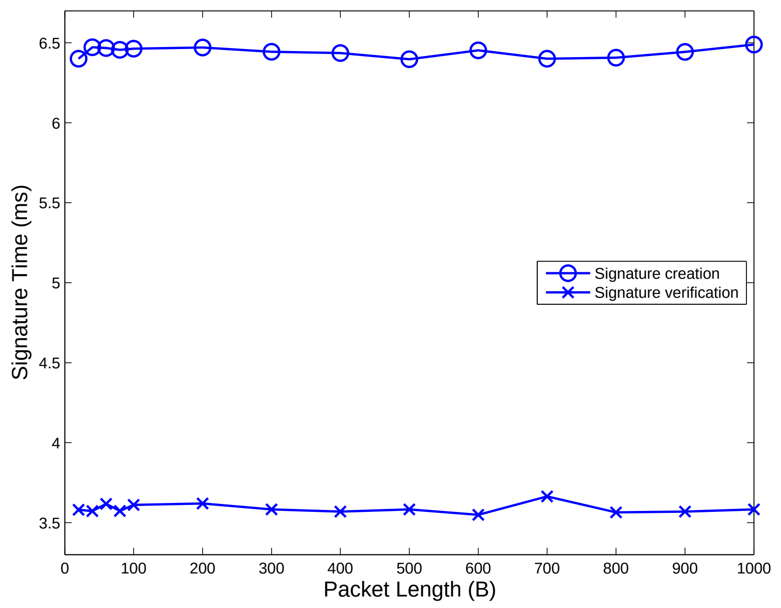

**Figure 6** Measured signature creation and verification times for packets of different lengths.

### *3.3 Availability*

Availability refers to the security requirement for information and network services to be available to their legitimate users, and may be affected if intermediate routers behave erratically (the case of hijacked nodes running attacker's software) or stop working, possibly due to a denial of service attack. Availability in CPN has been investigated previously in terms of the propagation of worms that disable nodes (Sakellari, 2011), denial of service attacks (Gelenbe and Loukas, 2004) and misbehaving routers (Lent and Gelenbe, 2012). Interestingly, as we will see, the self-adapting behaviour that is natural in CPN constitutes and inherent strength of the algorithm against all these threats.

#### *3.3.1 Misbehaving Routers*

The case of hijacked routers exhibiting an erratic behaviour has been recently studied (Lent and Gelenbe, 2012). However, this work did not introduce specific mechanisms to deal with router misbehavior. Two types of router misbehaviour were addressed: routers that drop a percentage of incoming packets and routers the direct smart packet to the worst possible next hop decision (according to the RNN algorithm).

Given that the reinforcement learning algorithm used in the RNN evaluates cumulative next hop behaviour, both types of misbehaviours can be addressed by introducing packet loss metrics in the formulation of routing goals as done in (Gelenbe et al., 2002) for normal packet drops. In particular, assuming a generic hop cost $C$ and estimations of the packet loss ratio $L$, the goal function for RNN training could be expressed as:

$$G = \frac{L}{1-L}P + C$$

where $P$ is an arbitrary penalty that is usually fixed to higher value than the cost of any path.

To experimentally evaluate this case, we have deployed a CPN testbed and conducted a number of trials and observations assuming that some of the core routers were compromised by an attacker, and therefore, had an abnormal behaviour.

A CPN implementation for the networking stack of the Linux kernel was used to carry out our tests. The implementation followed the algorithm described in Gelenbe (2004). The CPN implementation was deployed on 33 virtual machines, which were hosted by 6 quad-core physical machines using the VirtualBox hypervisor. These virtual machines allowed us to create a virtual topology, which was modeled after a real-world topology (ATT network). The exact technique that was employed to implement the virtual network can be found in the literature (Lent and Gelenbe, 2012). Each of the virtual machines consisted of a single core CPU with 256 KB of RAM and served to implement a software Linux router running the CPN module. The original ATT network topology that was available as a model for our virtual topology consisted of 27 nodes. However, 6 additional nodes were added to have extra communication endpoints, making a total of 33 nodes (and 50 links), of which 8 nodes had a single connection to the network and were used as endpoints for test traffic flows. While any network node could be attacked in a real network, we limited the simulated attacks to any of the 25 core routers given the scope of our study.

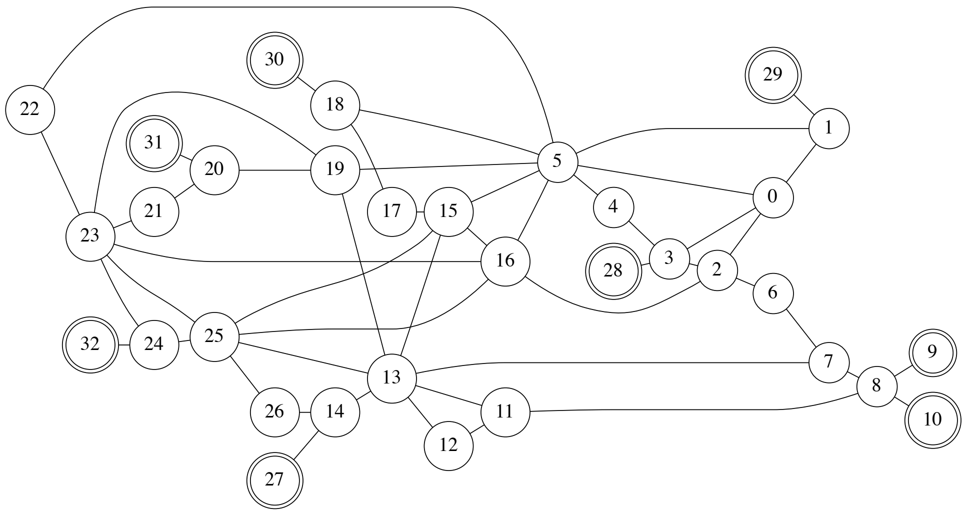

**Figure 7** A CPN testbed of virtual routers. A double circle indicates the source or sink of a test traffic.

At each endpoint, a traffic generator was installed to create UDP traffic at a fixed rate $\lambda$ with a given packet size $L$ and directed to another random endpoint, for a time period $T$. Rate $\lambda$ was randomly selected in the range 100–3000 Bps. Similarly, each flow duration $T$ was randomly assigned in the range from 1 to 3 minutes. Packets were assumed to be small (4B). The traffic generator created flows one after another so as to achieve an uninterrupted emission of packets from each endpoint during any experiment.

Smart packets were sent as a 0.1 fraction of normal packets to maintain path adaptation if needed in the case of a change in network conditions. At each hop, smart packet used a RNN to decide the next hop except for 10% of the decisions, which used a random neighbour selection that helped to facilitate network exploration and the discovery of alternative paths to destinations. Because of the use of virtualized resources in our tests, we opted for introducing smart packet routing based on router costs, which were randomly selected at the beginning of the experiment, in the range 1–100. For any given source-destination pair, smart packets had in general the task of finding the path with the minimum cost (the sum of the router costs involved in a path) or, if requested, the path with the minimum combination of cost and packet loss. The penalty value $P$ was fixed to 100 units.

Hijacked routers behaved maliciously by directing smart packets to the worst possible next hop as an attempt to increase the end-to-end flow cost or to prevent route discovery. Because CPN can easily isolate those routers, the difference of using the aforementioned goal functions to train the RNN was very small. Nevertheless, we did observe slightly higher loss ($10^{-4}$) ratio when only cost was used.

During the experiments, a number of core routers were assumed to be under the control of an attacker. Routers will drop a percentage of all the incoming normal traffic. Figure 8 depicts the measured ratio between the packet delivery rates for the case when the RNNs were trained using a cost-packet loss goal *Vs.* the case of RNNs only trained with a cost-based goal. It can be observed that as the number of hijacked routers increased, the comparative advantage of using a combined cost-loss goal diminished from about 20% to 5%.

### *3.3.2 Existing work on Resilience of CPN to worms*

Although CPN is generally very resilient to network changes, it was shown in (Sakellari, 2011; Sakellari and Gelenbe, 2009, 2010; Gelenbe, 2009) that it suffered worse performance during node failures, as the loss of acknowledgement packets led to insufficient training of the RNNs, the weights of which in a node are updated only when an ACK packet returns to it. For that reason, CPN nodes route a fraction of the SPs randomly, so that sudden changes of any kind could be discovered, but still in some scenarios it may need considerable numbers of random SPs before the decision of a node changes. The authors of (Sakellari, 2011; Sakellari and Gelenbe, 2009, 2010) investigated the performance of CPN during a worm propagation which resulted in network failures. The worm was spreading according to the AAWP (Analytical Active Worm Propagation) epidemiological model, a discrete-time and continuous state deterministic approximation model of the spread of active worms that scan for targets randomly (Chen et al., 2003).

A failure detection element was proposed to improve the performance of CPN in terms of packet delays and packet loss during failures. With this enhancement, at each RNN and for each neuron $i$, the timestamp of the last SP and the last ACK that used it, are stored. If no ACK has been received after sending the last SP then the link is considered “under failure” and the neuron corresponding to this link is

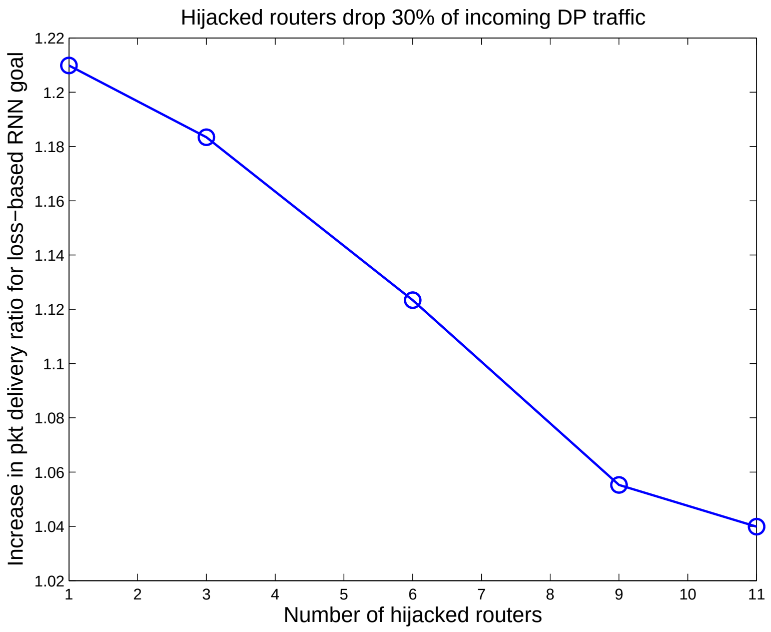

**Figure 8** Packet delivery improvement ratio of the RNN when trained with both cost and loss compared to only using cost.

considered "expired". Expired neurons do not participate in the calculation of the excitatory probabilities and the subsequent decisions of the RNN. The enhanced CPN was also compared with the Open Shortest Path First (OSPF) routing protocol and was shown to perform better in such situations.

### *3.3.3 Resilience to denial-of-service attacks*

Most protection systems for DoS attacks rely on hardware or software components that are added on top of an existing information and communication infrastructure (Loukas and Oke, 2010). Examples have been developed by Gelenbe et al. to achieve detection and response against such attacks. The first mathematical model for the analysis of the impact of flood-based DoS attacks was introduced in (Gelenbe and Loukas, 2004) and insights based on its numerical results led to the development of prototype defence implementations (Gelenbe and Loukas, 2005). These involved mechanisms for the selective dropping of packets based on their probability of being illegitimate, and were developed as modules of the CPN kernel. However, a significant weakness of defence mechanisms that are based on packet dropping is the substantial loss of valid packets due to false positives.

In (Gelenbe and Loukas, 2007), CPN's defence framework was improved with the introduction of a rate-limiting and prioritisation mechanism, where the result of validity tests against individual data packets determined the priority by which they would be served by the nodes. In recognition of the imperfection of any detection mechanism, by reducing the priority rather than dropping the packets that were detected to be malicious, false positives were reduced and the legitimate traffic served in the network was increased considerably.

Regardless of the specifics of the implementation of the response mechanism, the performance of a denial-of-service defence system is always highly dependent on the precision of detection. The detection approach presented in (Oke and Loukas, 2007) was based on the maximum likelihood principle and data fusion using the RNN, in both feedforward and recurrent architectures (Loukas and Oke, 2007). The input features used were the instantaneous bitrate and its rate of increase, the delay and its rate of increase, the entropy of the incoming traffic, as well as its Hurst parameter. In (Oke and Loukas, 2008), emphasis was placed on the distributed aspects of effective response.

In all these defence approaches, various forms of CPN were used as the underlying network infrastructure. We argue that the choice of the RNN-based CPN as the routing paradigm plays by itself a noticeable role in terms of the network's inherent resilience to flood-based availability attacks. Experiments presented in (Loukas, 2006) have indicated that CPN's dynamic routing improves considerably the performance of an imperfect DoS defence system during attacks of low to medium intensity (Fig. 9; $Pf$ and $Pd$ are the assumed defence system's probabilities of false positive and correct detection).

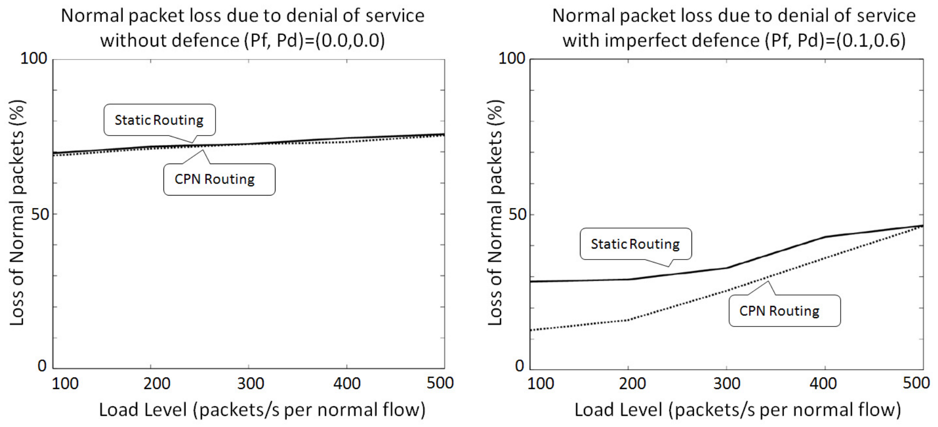

**Figure 9** Figure from (Loukas, 2006). Packet loss during a DoS attack for static and CPN dynamic routing, without DoS defence (left) and with an imperfect defence system (right)

## 4 Conclusions

The inherent self-adaptive properties of the Cognitive Packet Network constitute a significant advantage for helping it withstand attacks against its availability. Its self-adaptation becomes even more effective after introducing packet losses into the goal formulation for training the distributed random neural networks that are used in the routing algorithm. A denial of service defence mechanism with a given set of detection probabilities becomes more effective if applied in conjunction with the dynamic routing of CPN. Yet, similarly to the Internet Protocol, CPN was designed based on trust. By trusting the information carried by packets and that no node could be compromised, CPN becomes vulnerable to confidentiality and integrity attacks. It offers, by default, little support to ensure end-to-end confidential delivery of data or integrity of packets, which is of crucial importance given that it relies on real-time packets' information to effectively setup and maintain paths. Here, we have proposed extensions to the CPN protocol to explicitly address these weaknesses via the introduction of encryption and digital signatures. We have detailed the technical implementation of these mechanisms and argued that the benefits from strengthening the confidentiality and integrity, along with the availability of information transmitted through CPN, outweigh the disadvantage of the associated delays that are incurred.

## References

Chen, Z., Gao, L., Kwiat, K., Apr. 2003. Modeling the Spread of Active Worms. In: Proceedings of the IEEE INFOCOM 2003. Vol. 3. San Francisco, CA, USA, pp. 1890–1900.

Dhillon, G., Backhouse, J., 2000. Information System Security Management in the New Millennium. Communications of the ACM 43 (7), 125–128.

Gelenbe, E., Oct. 2004. Cognitive Packet Network. US Patent 6804201 B1.

Gelenbe, E., Jul. 2009. Steps toward self-aware networks. Commun. ACM 52 (7), 66–75.
URL `http://doi.acm.org/10.1145/1538788.1538809`

Gelenbe, E., Fourneau, J.-M., May 1999. Random neural networks with multiple classes of signals. Neural Comput. 11 (4), 953–963.
URL `http://dx.doi.org/10.1162/089976699300016520`

Gelenbe, E., Lent, R., Montuori, A., Xu, Z., Aug. 2000. Towards Networks with Cognitive Packets. In: Proceedings of the 8th International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (IEEE MASCOTS). San Francisco, CA, USA, pp. 3–12, opening Invited Paper.

Gelenbe, E., Lent, R., Montuori, A., Xu, Z., Oct. 2002. Cognitive packet networks: QoS and performance. In: Proceeedings of The Tenth IEEE/ACM International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS). Ft. Worth, Texas, pp. 3–12, opening Keynote Paper.

Gelenbe, E., Lent, R., Xu, Z., Oct. 2001a. Design and Performance of Cognitive Packet Networks. Performance Evaluation 46 (2-3), 155–176.

Gelenbe, E., Loukas, G., 2007. A Self-Aware Approach to Denial of Service Defence. Computer Networks 51 (5), 1299–1314.

Gelenbe, E., Seref, E., Xu, Z., Feb. 2001b. Simulation with Learning Agents. Proceedings of the IEEE 89 (2), 148–157.

Gelenbe, E., Xu, Z., Seref, E., Nov. 1999. Cognitive Packet Networks. In: Proceedings of the 11th International Conference on Tools with Artificial Intelligence (ICTAI '99). IEEE Computer Society Press, Chicago, IL, USA, pp. 47–54.

Gelenbe, E., G. M., Loukas, G., October 2004. Defending Networks against Denial of Service Attacks. In: in Proceedings of the Conference on Optics/Photonics in Security and Defence, Unmanned/Unattended Sensors and Sensor Networks. Vol. 5611. London, UK, pp. 233–243.

Gelenbe, E., G. M., Loukas, G., June 2005. An Autonomic Approach to Denial of Service Defence. In: in Proceedings of the IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks. Taormina, Italy, pp. 537–541.

Gellman, M., Mar. 2007. Quality of service routing for real-time traffic. Ph.D. thesis, Imperial College London, London, UK.

Lent, R., Gelenbe, E., Oct. 2012. Evaluating cpn on a large virtual network testbed. In: Proceedings of the 27th International Symposium on Computer and Information Sciences Series: Lecture Notes in Electrical Engineering. Paris, France.

Loukas, G., 2006. Defence against denial of service in self-aware networks, PhD thesis, University of London, London, United Kingdom.

Loukas, G., Oke, G., July 2007. Likelihood ratios and recurrent random neural networks in detection of denial of service attacks. In: Proceedings of International Symposium of Computer and Telecommunication Systems. Vol. 7. San Diego, CA, USA.

Loukas, G., Oke, G., 2010. Protection Against Denial of Service Attacks: A Survey. The Computer Journal 53 (7), 1020 – 1037.

Oke, G., Loukas, G., 2007. A Denial of Service Detector based on Maximum Likelihood Detection and the Random Neural Network. The Computer Journal 50 (6), 717 – 727.

Oke, G., Loukas, G., September 2008. Distributed Defence Against Denial of Service Attacks: A practical view. In: in Proceedings of the BCS International Conference on Visions of Computer Science. London, UK, pp. 153–162.

Sakellari, G., June 2009. The Cognitive Packet Network: A Survey. The Computer Journal: Special Issue on Random Neural Networks, doi:10.1093/comjnl/bxp053.

Sakellari, G., 2011. Performance evaluation of the Cognitive Packet Network in the presence of network worms. Performance Evaluation 68 (10), 927–937.

Sakellari, G., Gelenbe, E., May 2009. Adaptive Resilience of the Cognitive Packet Network in the presence of Network Worms. In: Proceedings of the NATO Symposium on C3I for Crisis, Emergency and Consequence Management. Bucharest, Romania, pp. 16:1–16:14.

Sakellari, G., Gelenbe, E., 2010. Demonstrating cognitive packet network resilience to worm attacks. In: ACM Conference on Computer and Communications Security. pp. 636–638.