Advanced topics in Distributed Systems

group assignment topics

spring 2011

Contents

A security experimentation framework within NS-3

Supervisors: Sebastien Mondet (smondet@ifi.uio.no)

Description:

The goal of this project is to provide a toolkit for running NS-3 MANET-based scenarios containing misbehaving nodes. In other words, we need NS-3 modules implementing specific attacks allowing us to inject "attackers" in simulation scenarios. We expect the students to implement and validate various NS-3 modules. For example, one could implement one or more modified versions of NS-3's OLSR module which execute known attacks of the protocol [1]. Attacks concerning "on-demand" protocols are also needed [2]. The delivered NS-3 modules should be validated through proof-of-concept simulations showing the success of the attacks. They should also be easily integrable in existing simulations.

Required competences: C++, NS-3 (a tutorial will be given).

References:

[1] Wang, M., Lamont, L., Mason, P., and Gorlatova, M. 2005. An effective intrusion detection approach for OLSR MANET protocol. 1st IEEE ICNP Workshop on Secure Network Protocols (NPSec). 1st IEEE ICNP Workshop on, pp. 55-60.

[2] Hu, Y., Perrig, A., and Johnson, D. 2003. Rushing attacks and defense in wireless ad hoc network routing protocols. In proceedings of the 2nd ACM workshop on Wireless security.

Delay Tolerant Epidemic Streaming

Area: Delay tolerant networking, multimedia streaming.

Supervisors: Daniel Rodriguez Fernandez (dani@ifi.uio.no)

Description:

The Delay Tolerant Networking (DTN) [1] is networking paradigm that is applied to challenged networks, which might present long delays amongst the nodes that compose them. This paradigm can be applied to many network instances, such as the Interplanetary networking, sparse Mobile Ad Hoc Networks (MANETs), wireless sensor networks, etc. Such networks might be subject to disruptions where different network partitions become out of communication range of each other. It is even possible that there does not exist at any time a direct path of communication between two nodes. In order to cope with these challenges there have been proposed different solutions that tried to exploit the mobility of the nodes, which may act as data carriers between different partitions. This behaviour follows the store-carry-forward paradigm, where a node stores the data, carries it for a while, and finally, when new routes become available, forwards it. Note that this is opposed to the behaviour of the routers on the Internet where, as there is assumed end-to-end connectivity amongst all the nodes, the packets are dropped if there is no route to reach the destination. The aim of this assignment is to study the possibility of adapting existing DTN systems such as Epidemic Routing (ER) [2] for multimedia streaming. The original ER was designed for the message services (e-mail like). Therefore, there should be taken into account the differences introduced by the multimedia streaming and propose modifications to the classical ER. Examples of potential optimisations could be extending the resume messages with information about the streams, gossiping (third nodes listens to other nodes communications), packet priorities or congestion control during the message interchange phase. The students are also encouraged to propose and implement their own streaming optimisation ideas. For testing and evaluation, the designed system should be implemented as an application in the ns-3 [3] network simulator. In a first phase classical ER would be implemented and used as base for the streaming extensions.

Tasks:

1. Study the challenges that multimedia streaming present for classical ER systems.

2. Design extensions for ER where streaming challenges are handled.

3. Implementing ER and its extensions as applications into the ns-3 simulator.

4. Test of the correctness of the DTN Epidemic Streaming and compare its performance with regular ER when forwarding multimedia streams.

Required competences: C/C++ language is required. A tutorial about how to use and implement applications in ns-3 will be given.

References:

[1] K. Fall, A Delay-Tolerant Network Architecture for Challenged Internets, IRB-TR-03-003, 2003

[2] A. Vahdat, D. Becker, Epidemic Routing for Partially-Connected Ad Hoc Networks, 2000

[3] ns-3, The ns-3 network simulator, `http://www.nsman.org/`

Identity Sub-layer: Separating Identify from Location

Area: Future Internet

Supervisors: Daniel Rodriguez Fernandez (dani@ifi.uio.no)

Description:

Today's modern handheld devices have an increasing number of alternatives for wireless communicating, such as 802.11, 802.16, UMTS, GPRS or Bluetooth. However, such multi-homing in the wireless interface technology cannot be easily exploited because of the Internet Protocol (IP) [1], which does not support multi-homing. In the same way, IP does not provide native support node mobility, which occasionally may require updating the point of attachment with the Internet, and therefore its IP address. This may cause the breakage of the communication sessions that involve the mobile node. The lack of support of both mobility and multi-homing comes from some initial decisions in the design of the Internet, such the dual role that plays the IP address, which acts as locator for routing, but also as node identifier. In the original design of IP, it was assumed that each host had only one network interface. This was associated with an IP address, which had topological meaning. The dual role of the IP address was a shortcut, which in the long term has limited the evolution of the core of the Internet, since the three upper layers are attached to the IP addresses. If an ID-Locator split is performed, multi-homing and mobility support would become easier, and at the same time it may be possible to use other network technologies, such as for instance IPv6, since the application layers would not be attached any more to an IP address but to a host identifier. The aim of this assignment is to study the different alternatives for ID-Locator split, and design and implement an Identity sub-layer, which enables the ID-Locator mapping. The design of this new sub-layer should be flexible enough to allow the integration of different network layers, such as IPv4, IPv6, or other future networking protocols. For simplicity this sub-layer would be initially implemented as an ns-3 [2] application above the transport layer. It should be able to handle node mobility and node multi-homing. As an optional extension, the Identity sub-layer could be implemented as a layer between transport and

network layers. In this case the application and transport layers would use host identifiers, while the network would use network locators (for example IP addresses). It is possible to find different proposals for the ID-Locator split in the bibliography, such as the Host Identity Protocol [3] (HIP) or the Location Independent Network Architecture (LINA) [4].

Tasks:

1. Understand the problems that IP has to handle mobility and multi-homing.
2. Study different approaches for the Identifier-Locator split.
3. Design an Identity sub-layer, which should perform the mapping between identifiers and locators.
4. Implementing a network application for the ns-3 simulator, which provides the functionality of the Identity sub-layer.
5. Test of the correctness of the Identity sub-layer.
6. (Optional) Implement the Identity sub-layer as a layer that separates Transport and Network layers (for IPv4 or IPv6).
7. (Optional) Make that the Identity can work with both IPv4 and IPv6 at the same time.

Required competences: C/C++ language is required. A tutorial about how to use and implement applications in ns-3 will be given.

References:

[1] RFC 791, The Internet protocol, September 1981
[2] NS-3, The ns-3 network simulator, `http://www.nsman.org/`
[3] P. Nikander, A. Gurtov, T. Henderson, Host Identity Protocol (HIP): Connectivity, Mobility, Multi-homing, Security, and Privacy over IPv4 and IPv6 networks, to appear in IEEE Communications Surveys and Tutorials, second quarter, 2010.
[4] Kunishi, M. and Ishiyama, M. and Uehara, K. and Esaki, H. and Teraoka, F. LIN6: A New Approach to Mobility Support in IPv6. International Symposium on Wireless Personal Multimedia Communication, 2000.

Policy Based Network Selection for Multi-homed Nodes

Area: Future Internet

Supervisors: Daniel Rodriguez Fernandez (dani@ifi.uio.no)

Description:

Today's modern handheld devices have an increasing number of alternatives for wireless communicating, such as 802.11, 802.16, UMTS, GPRS or Bluetooth. However, the lack of native support for multi-homing in IP makes difficult to take advantage of it. This assignment aims to study how policy based systems can be applied to multi-homed nodes. The students should design and implement a system that selects the best network alternative for given a set of policies. Some example of rules may be: "The best is the network with minimum latency", "The packets of type X should be send through the network with minimum packet loss", "Ensure a minimum bandwidth of X Kbps", etc. For supporting the decisions a monitoring facility should be designed and implemented. The policies should be evaluated dynamically to adapt their decisions to the updates on the monitoring information. The monitoring facility will collect that information from monitors, which will be specialised on measure a specific metric, such as for example latency. There should be defined interfaces for policies and for monitors, so the system can be easily extended. In order to test and evaluate the components it will be implemented an application for the ns-3 [1] simulator, where the students should test the effect of applying different policies and monitors. For simplicity, in this assignment it will be assumed that a multi-homed node is member of different networks, which are independent and not interconnected. Thus, the networks can be seen as point-to-point links between the nodes.

Tasks:

1. Design and implement the policy based network selection component, the monitoring facility component. It is important that they could be extended easily.

2. Design the interfaces of the policies and the monitors. Implement some

instances of policies and the monitors that are required by these policies.

3. Implement an application for ns-3 that should run on two nodes, which will be connected through two different networks of the simulated environment that is able to use the implemented components to select the best network alternative.

4. Test and evaluate of the policy based routing.

5. (Optional) create policies for multimedia streaming.

6. (Optional) increase the complexity of the scenario adding more networks and nodes.

Required competences: C/C++ language is required. A tutorial about how to use and implement applications in ns-3 will be given.

References:

[1] NS-3, The ns-3 network simulator, `http://www.nsman.org/`

Studying the performance impact between processing and forwarding

Supervisors: Stein Kristiansen (steikr@ifi.uio.no)

Background:

In DT-Stream, we are developing delay-tolerant streaming solutions for Mobile ad-hoc networks (MANETs) for rescue and emergency operations. In these scenarios, the MANET is formed by devices carried by on-site rescue team members. Since they need to be able to move freely and unconstrained, the carried devices must be small and portable. When used as intermediate nodes, it has been shown that the limited resources on such devices can have a significant impact on forwarding performance [1,2].

Description:

Since devices are required to function both as routers and communication end-points, they are in many cases required to perform several resource-intensive tasks at the same time. Furthermore, since MANET solutions are often implemented as user-space overlays/middlewares, the computational overhead per forwarded byte and packet is usually higher than in-kernel protocol handling. In this assignment, the students should investigate how varying different types of computational overhead affects application and forwarding performance. Example tasks include:

1. How concurrent application load affects forwarding performance
2. How forwarding load affects application performance
3. How moving the forwarding from kernel-space to a user-space application affects forwarding performance.
4. How varying user-space per-packet and per-byte processing affect forwarding and application performance.

Test-bed: The students will perform real-world experiments with the recent Nokia N900 smartphone, embedded into a two-hop wireless ad-hoc network test-bed. This test-bed should consist of the N900 forwarding streams between a source and a destination laptop. The final results should however

be obtained from runs in a shielded environment, e.g., in the bomb-shelter at IFI1. For convenience, however, setting up the test-bed and ensuring correct execution of the experiments can be performed in areas with external WLAN interference, e.g., in the student labs.

Workload: Simple and generic processing and traffic workload should be utilized to obtain generic and repeatable results. For processing, e.g., varying numbers of simple threads, each consisting of e.g., a simple while-loop could be used. For traffic, uniform UDP traffic streams, with variable packet-size and -rate can be generated with traffic generators such as RUDE/CRUDE [3]. For task nr. 2 above, the students are required to implement minimal user-space forwarding.

Measurements: The students should obtain end-to-end performance measurements, including throughput and jitter (and delay if possible), as well as node-resource utilization, such as CPU utilization and buffer utilization.

Summary of Tasks: Throughout the course of the project, the students should perform the following tasks:

1. Read the background material to understand the research problem
2. Set up, configure and test the test-bed
3. Design the experiments and workload application
4. Prepare the necessary tools, and write the experiment scripts
5. Perform the experiments
6. Analyze the results

Required competences: Linux, Networking, C programming and scripting, Some general knowledge on common OS mechanisms and hardware

References:

[1] M. Halvorsen, T. Plagemann, and M. Siekkinen. Video Streaming Over MANETs: Reality or Fiction? In MobiMedia '08, Proceedings of the 4th International Mobile Multimedia Communications Conference, 2008.

[2] S. Kristiansen, M. Lindeberg, D. Rodriguez-Fernandez, and T. Plagemann. On the forwarding capability of mobile handhelds for video streaming over manets. In Proceedings of the second ACM SIGCOMM workshop on

Networking, systems, and applications on mobile handhelds (MobiHeld '10), pages 33–38. ACM, 2010.
[3] RUDE/CRUDE: http://rude.sourceforge.net/

Investigating the impact of power-saving on DT-Stream nodes' performance

Supervisors: Stein Kristiansen (steikr@ifi.uio.no)

Background:

In DT-Stream, we are developing delay-tolerant streaming solutions for Mobile ad-hoc networks (MANETs) for rescue and emergency operations. In these scenarios, the MANET is formed by devices carried by on-site rescue team members. Since they need to be able to move freely and unconstrained, the carried devices must be small and portable. When used as intermediate nodes, it has been shown that the limited resources on such devices can have a significant impact on forwarding performance [1,2].

Description:

Small, portable devices have small batteries, and therefore are strictly energy constrained. State-of-the-art devices are therefore equipped with sophisticated power-management mechanisms that lower CPU- and bus-frequencies to save energy. In this assignment, the students will perform real-world experiments with a recent hand-held device to investigate how varying the power-saving mode affects the performance of central operations performed by a DT-Stream node. These operations include:

1. 2-hop IP-forwarding of streams: unveils the kernel-space forwarding performance of the device working as an intermediate router.

2. 1-hop transmission of data from an application: unveils the performance of the device when acting as a source or carrier.

3. 1-hop reception of streams at an application: unveils the performance of the device acting as a carrier or destination.

Test-bed: The students will perform real-world experiments with the recent Nokia N900 smartphone, embedded into a two-hop wireless ad-hoc network. This test-bed should consist the N900 and laptops. For convenience, setting up the test-bed and ensuring that the experiments run correctly can be performed in areas with external WLAN interference, e.g., in the student

labs. The final results should however be obtained from runs in a shielded environment, e.g., in the bomb-shelter at IFI1.

Traffic Workload: Simple and generic traffic workload should be used to obtain generic and repeatable results. For this, a traffic generator (e.g., RUDE/CRUDE [3]) should be used to inject uniform UDP traffic streams. If time permits, the students can in addition investigate the performance TCP-transmission.

Factors: Packet-size, packet generation rate and power management parameters (see below). If there is time for experiments with TCP, the number of transmissions can be varied.

Power Management Parameters: The N900 runs on the Linux-based Maemo 5 platform. Selecting different power-saving modes can be done by varying CPU frequency, and/or the QoS requirements of the platform. The latter can be achieved by deploying a simple kernel-module using the power-management QoS interface provided by the kernel. This module will be provided by the supervisor, allowing the request of minimum network throughput, and maximum network and CPU-to-DMA latencies.

Measurements: The students should obtain end-to-end performance measurements, including throughput and jitter (and delay, if time), as well as node-resource utilization, such as CPU and buffer utilization (and intra-node packet handling delays, if time).

Summary of Tasks: Throughout the course of the project, the students should perform the following tasks:

1. Read the background material to understand the research problem
2. Set up, configure and test the test-bed.
3. Design the experiments
4. Prepare tools and implement the necessary experiment scripts
5. Perform the experiments
6. Analyze the results

Required competences: Linux, Networking, C programming and scripting, Some general knowledge on common OS mechanisms and hardware

References:

[1] M. Halvorsen, T. Plagemann, and M. Siekkinen. Video Streaming Over MANETs: Reality or Fiction? In MobiMedia '08, Proceedings of the 4th International Mobile Multimedia Communications Conference, 2008.

[2] S. Kristiansen, M. Lindeberg, D. Rodriguez-Fernandez, and T. Plagemann. On the forwarding capability of mobile handhelds for video streaming over manets. In Proceedings of the second ACM SIGCOMM workshop on Networking, systems, and applications on mobile handhelds (MobiHeld '10), pages 33–38. ACM, 2010.

[3] RUDE/CRUDE: `http://rude.sourceforge.net/`

The communication load of individual nodes in MANETs

Supervisors: Stein Kristiansen (steikr@ifi.uio.no)

Background:

In DT-Stream, we are developing delay-tolerant streaming solutions for Mobile ad-hoc networks (MANETs) for rescue and emergency operations. In these scenarios, the MANET is formed by devices carried by on-site rescue team members. Since they need to be able to move freely and unconstrained, the carried devices must be small and portable. When used as intermediate nodes, it has been shown that the limited amount of computational resources on such devices can have a significant impact on end-to-end performance [1,2].

Description:

Although network simulators today typically do not model nodes resources, the results can be used to obtain an indication of how much communication load devices should be able to handle once deployed. In this assignment, the students should simulate (using ns-3) streaming over wireless ad-hoc networks with different typology and mobility patterns, and obtain, parse and analyse the communication load of individual nodes throughout the simulation. The results should be helpful to determine the minimum requirements of devices deployed in these scenarios. The students should finally investigate published results on the capabilities of modern handhelds [1,2] and discuss the probability of these devices being powerful enough to handle the simulated scenarios.

Interesting Topologies:

1. Chain-topology: The goal is to unveil how nodes' distances and the number of hops between a source and destination affect the forwarding load of intermediate nodes.

2. Sparse and dense MANETs with random mobility: The goal is to investigate what the requirement of intermediate nodes is in random mobility scenarios of varying density.

3. Delay tolerant streaming scenarios: Store-carry-forward operations may temporarily impose high communication load in situations where car-

riers empty their buffers as fast as possible. Using the delay-tolerant streaming simulation framework (DTS) developed in the DMMS group, the students should simulate delay-tolerant streaming in realistic emergency and rescue scenarios.

Factors: Both the structure, dynamicity and capacity of the network, as well as the amount of injected traffic workload, will affect results. For Scenario 1 above, the number of and distance between nodes, the amount of injected traffic and the capacity of the medium can be varied. In Scenarios 2 and 3, the velocity and waiting time between each movement and the size of the area can in addition be varied.

Measurements: The students should measure the bitrate of transmission, forwarding and reception during complete simulation runs of all involved nodes. If time, the students should also obtain energy consumption on each node using the energy model available in ns-3.

Summary of Tasks: Throughout the course of the project, the students should perform the following tasks:

1. Read the background material to understand the research problem
2. Design the experiments
3. Implement the ns-3 and shell scripts for experiment automation
4. Perform the experiments
5. Analyze the results

Required competences: Networking, Scripting (to be able for parse traces), Some C++ programming knowledge

References:

[1] M. Halvorsen, T. Plagemann, and M. Siekkinen. Video Streaming Over MANETs: Reality or Fiction? In MobiMedia '08, Proceedings of the 4th International Mobile Multimedia Communications Conference, 2008.

[2] S. Kristiansen, M. Lindeberg, D. Rodriguez-Fernandez, and T. Plagemann. On the forwarding capability of mobile handhelds for video streaming over manets. In Proceedings of the second ACM SIGCOMM workshop on Networking, systems, and applications on mobile handhelds (MobiHeld '10), pages 33–38. ACM, 2010.

A Ping-Pong Based Approach For Packet-loss Prevention in Disruptive MANETs Using The Dts-Overlay

Supervisors: Morten Lindeberg (mglindeb@ifi.uio.no)

Description:

When working with the Dts-Overlay over disruptive MANETs, we found that a high amount of packet-loss could be attributed the fact that the routing protocol (OLSR) reported routes that in fact included links that was down [1]. As we are working with video streams rather than file transfer we do not rely on end-to-end feedback. Thus TCP is not preferred as a measure of transport. Without feedback, packets are lost. When working with a solution to resolve another problem, that is MAC address resolution, we found that by doing some kind of ping-pong based test in advance of sending data over a newly found link, that is, postponing the use of it until we get the reply in the form of a pong, we would reduce packet loss.

In this assignment, we want to extend somehow the idea of these ping-pong mechanisms. To ease development, we propose to use the existing Dts-Overlay, a framework for disruptive MANET development, which currently runs in the ns-3 [2] network simulator. More specifically, implement a control protocol for testing links through simple hop-by-hop ping-pong functionality. Further more, see what we can get out of e.g., statistics about the round-trip time (RTT), pong loss rate, etc., in terms of identifying network state, and possible adapt to this.

Tasks:

1. Understand the Dts-Overlay, and how to run simulations in ns-3.
2. Understand existing efforts for limiting packet loss/adding reliability in challenged networks.
3. Design and implement the ping-pong functionality, measure statistics online, and adapt to the link state.
4. Evaluate performance, e.g., comparing with TCP.

Required competences: Network protocols and C++ programming.

References:

[1] Lindeberg, M., Kristiansen, S., Goebel, V. and Plagemann, T. MAC Layer Support for Delay Tolerant Video Transport in Disruptive MANETs, to appear in IFIP/TC6 NETWORKING 2011.
[2] The ns-3 network simulator, `http://www.nsnam.org/`

Minimal-Intrusive Rate Control For Disruptive MANETs

Supervisors: Morten Lindeberg (mglindeb@ifi.uio.no)

Description:

The classical approach for rate control on the Internet today, primarily "TCP congestion avoidance algorithm", relies on end-to-end feedback. TCP does not work well over disruptive MANETs. For one, the random packet loss caused by the wireless medium might be mistaken as congestion. Second, end-to-end delays might lead to TCP mechanisms timing out the connection, or causing that a connection is never made. Also, for the purpose of video streaming, as opposed to file transfer, TCP has drawbacks. While its purpose is to provide reliable byte transfer, it typically fragment packets into a byte stream, instead as for instance RTP+UDP based video transfer, where it is possible to maintain encoded video frames per packet for optimally meeting delay constraints. For the purpose of unicast streaming, e.g., video or telephony, the TCP Friendly Rate Control (TFRC) [1], is said to lower the variation of throughput, compared to TCP, however it still relies on end-to-end feedback mechanisms. With the strict principles of layering in mind, such a mechanisms, i.e., exchange of control messages used to determine the state of the network, is the only way of determining an appropriate rate. However, if we allow for cross-layer parameter exchange, it might be that we can minimize network intrusion, i.e., significantly lower the amount of needed control packets. Instead, we can rely on the state of lower protocols, for instance the MAC sub-layer of the IEEE 802.11a/b/g WiFi protocol.

In this assignment, students are supposed to develop a rate control algorithm relying mostly on cross-layer parameter exchange, such as the retransmission queue length on the MAC sub-layer. To ease development, we propose to use an existing framework for disruptive MANET development, called Dts-Overlay, which currently runs in the ns-3 [2] network simulator. Currently, a simple algorithm for determining the rate of which to empty so-called store-carry-forward buffers exist, and can be used as a starting point.

Tasks:

1. Understand the Dts-Overlay, and how to run simulations in ns-3.

2. Get an overview of existing rate control algorithms, TCP, TFRC, etc.

3. Design and implement at least one rate control algorithm, either for end-to-end transport, or just, as is the case with the buffer empty algorithm, for per-hop data transfer along the route.

4. Evaluate performance of the control algorithm.

Required competences: Network protocols and C++ programming

References:
[1] TCP Friendly Rate Control (TFRC): Protocol Specification, `http://www.faqs.org/rfcs/rfc3448.html`
[2] The ns-3 network simulator, `http://www.nsnam.org/`

Energy Balanced Node Forwarding In Dts-Overlay

Supervisors: Morten Lindeberg (mglindeb@ifi.uio.no)

Description:

Mobile handheld devices have limited battery capacity, and data transmission is known to drain battery. In mobile ad hoc networks (MANETs), nodes take the role as forwarders, i.e., the role that designated routers have on the Internet. During the process of performance evaluation, limited battery life is often omitted. The objective of this task is to further study the impact that limited battery life has on performance in MANET networking.

Part of ongoing work, the Dts-Overlay can serve as a framework for developing strategies for delay tolerant video streaming in disruptive MANETs. The framework is set up for simulation studies, using the ns-3 network simulator [1]. Ns-3 currently includes a component that models node energy consumption based upon the amount of data transmissions locally. To be more specific, this assignment asks for the students that they incorporate measurements from the energy model for each node into the "Resource Management" component, part of the Dts-Overlay. This component monitors node network protocol statistics across the simulated network layers of the each node locally. In addition, students should make sure those nodes, as they are drained of energy to the point that their battery is empty, is removed from the network. As a final step towards mastering this assignment, students should try to incorporate at least one strategy in the Dts-Overlay "Decision Make" component, that in the process of deciding next hop for the purpose of packet forwarding along known network routes, takes node energy consumption into consideration.

Tasks:

1. Understand the Dts-Overlay, and how to run simulations in ns-3.
2. Understand the ns-3 energy module.
3. Survey existing work on energy consumption and MANET networking.
4. Incorporate measurements from nodes energy consumption locally, into the Dts-Overlay

5. Make sure that nodes "go down" as their battery is drained.

6. Design at least one strategy that tries to improve the overall lifetime of nodes.

7. Evaluate performance of this strategy, and present measured results with and without the energy model.

Required competences: Network protocols and C++ programming

References:

[1] The ns-3 network simulator, `http://www.nsnam.org/`

Cooperative Multiple Camera for Surveillance System

Supervisors: Viet Hoang Nguyen (viet@ifi.uio.no)

Description:

Camera Surveillance is becoming increasingly common place but it is practically impossible for anybody to monitor continuously what is happening inside these surveyed area. Hence we need automated surveillance mechanisms which can understand what is happening in the scene and automatically react to it. Typically, one sensor is not enough to undertake the entire task hence we need multiple sensors which are required to cooperate to collectively perform the surveillance task. This brings us to the use of active, cooperative, multi-sensor environments.

Consider a triple camera surveillance scenario where system goal is to obtain high resolution images of an person entering an area of interest. In this scenario, based on the distance from person to camera, the angle of face ..., three cameras cooperate towards different goals which are object detection, face detection and face recognition. We expect three camera will achieve much more accurate and also solve the problem that single one does not in these task. Using existing algorithm provided from some live face recognition library for example Verilook [2], we also looking for a multi-camera solution to distinguish real face with face in static image and even face in video played on mobile phone or monitor.

Tasks:

1. Deploying camera, optimizing the position to cover area of interest which we expected to recognize person.
2. Programming a software which control pan tilt camera to work together in three tasks object detection, face detection and face recognition.
3. Conduct experiments, and analyse measurements.

Required competences:

- Having advance skill with a high level programming language. (C#, C++ are preferred)
- Having experience with image processing and motion sensor (not strictly required)

Good luck and have fun with our team.

References:

[1] Panasonic Camera. http://www2.panasonic.com/consumer-electronics/support/Computers-Networking/Network-Cameras/Network-Cameras/model.BL-C131A

[2] VeriLook SDK Face Identification Library. http://www.neurotechnology.com/verilook.html

[3] Liang Liu, Huadong Ma, and Xi Zhang. On directional k-coverage analysis of randomly deployed camera sensor networks. In Communications, 2008. ICC '08. IEEE International Conference on, pages 2707 –2711, May 2008.

[4] Siva Ram, K. R. Ramakrishnan, P. K. Atrey, V. K. Singh, and M. S. Kankanhalli. A design methodology for selection and placement of sensors in multimedia surveillance systems. In Proceedings of the 4th ACM international workshop on Video surveillance and sensor networks, VSSN '06, pages 121–130, New York, NY, USA, 2006. ACM.

[5] Vivek K. Singh and Pradeep K. Atrey. Coopetitive visual surveillance using model predictive control. In Proceedings of the third ACM international workshop on Video surveillance sensor networks, VSSN '05, pages 149–158, New York, NY, USA, 2005. ACM.

[6] Panasonic Network Camera CGI Specifications. Camera cgi interface v3.51 (find on internet).

FindMe robot

Supervisors: Azadeh Abdolrazaghi (azadeha@ifi.uio.no)

Have you ever thought how we can get help from robots in our daily life? You may say that is an imagination like science-fiction movies but I am sure you can make it real today.

Suppose your grandparents live alone somewhere far from all the relatives. They have got dementia and/or poor vision and they cannot find their way around their house. They may go in a wrong way, hit some obstacles, hurt themselves and at the end forget what they want to do. Would you like to make their life easier and prevent them from more injuries and pain?

You can try helping them by programming a LEGO Robot. It may seem impossible but you can try. People have invented very useful instruments out of very primitive stuff while this robot is not primitive at all and has several sensors and actuators.

Description:

We have two LEGO mindstorms robots available in our lab which you can program to make a navigator robot. They have 4 sensors: touch, microphone, ultrasonic and light. They also have 3 actuators (servo motors) for movement and grabbing objects. Sensors can be used to recognize the environment, obstacles, distance to and color of them. You need to carefully study the capabilities of the sensors and actuators to design a very good algorithm taking benefit of all the capabilities as much as possible. Understanding the capabilities of your robot, you can specify the requirements and limitations of your system. You should choose an appropriate user interface to guide the person. You can send/receive files to/from robot using USB and Bluetooth.

Tasks:

First you need to design an efficient navigation algorithm considering the available sensors. And then implement your ideas on robot using the software that comes with it, supporting both Windows and Mac. The software has a

graphical interface and uses LabView. For more advanced programming you can upload LeJOS operating system on the robots and program them using Java.

Required competences: No special skills is required except passion, motivation and hard working.

References:

[1] Manual of LEGO mindstorms (2 hard copies available)
[2] Official website of LEGO mindstorms: `http://mindstorms.lego.com/en-gb/Default.aspx`
[3] LEGO Mindstorms blog: `http://thenxtstep.blogspot.com/`
[4] LeJOS, Java for LEGO mindstorms: `http://lejos.sourceforge.net/`

Message Ferry for Emergency and Rescue Operations

Supervisors: Daniel Rodriguez Fernandez (dani@ifi.uio.no), Francisco Velazquez (francisv@ifi.uio.no), Hans Vatne Hansen (hansvh@ifi.uio.no)

Description:

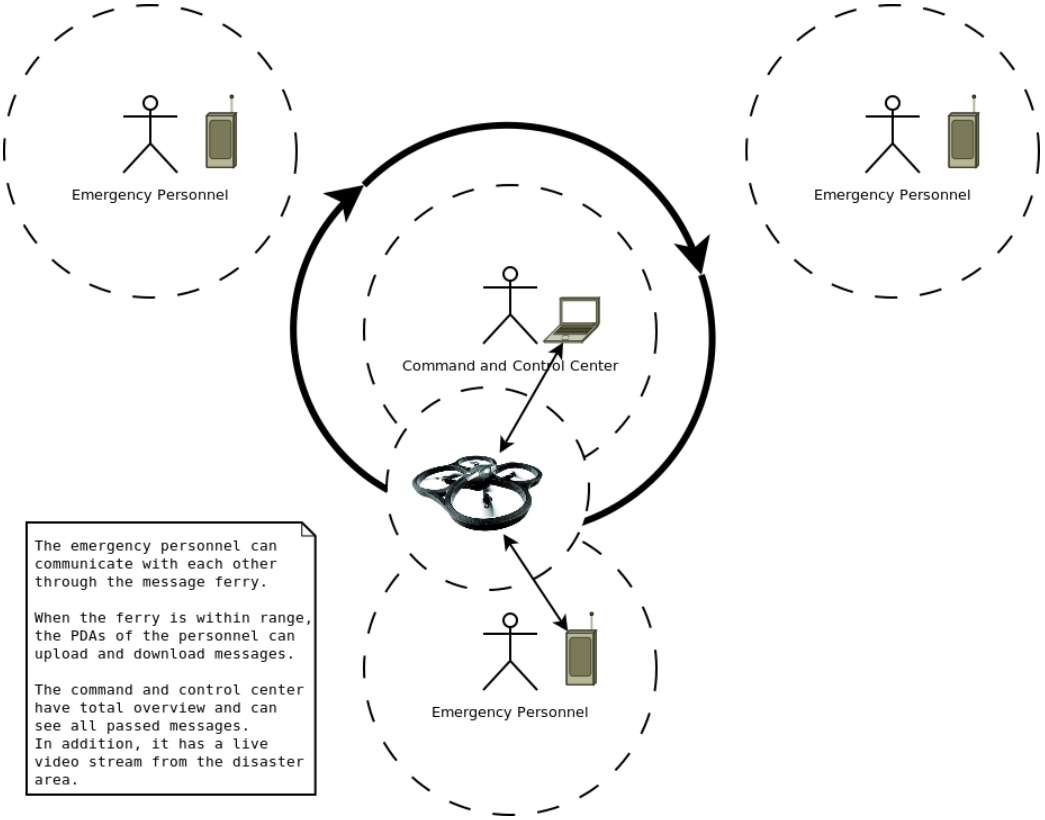
In emergency and rescue scenarios a Command and Control Center (CCC) is responsible for managing doctors, firemen and other personnel in the disaster area.

The infrastructure in such areas can be unreliable or even destroyed. In the DT-Stream project we build our own ad-hoc networks and pass messages and information between the involved parties in a delay tolerant fashion.

In this assignment you will develop a message ferry for passing messages between the involved parties. You will use a Laptop for sending commands and receiving video, and an AR Drone as Message Ferry and camera.

(a) Command and Control Center (aka Laptop)

(b) Message Ferry – AR Drone



Tasks:

1. Develop a static path for the AR Drone to patrol. Send video back to

CCC.

2. Enable receiving and sending messages from the personnel within WiFi range.
3. Make CCC give dynamic paths to the message ferry.

Required competences:

- programming: C and Bash script,
- tools: Linux/BusyBox on the Drone, Telnet and FTP.

References:

- AR Drone API: `https://projects.ardrone.org/wiki/ardrone-api/`
- AR Drone Developer Guide (PDF)
- Wenrui Zhao, Mostafa Ammar, Ellen Zegura: A Message Ferrying Approach for Data Delivery in Sparse Mobile Ad Hoc Networks, 2004. The Fifth ACM International Symposium on Mobile Ad Hoc Networking and Computing.

Mobile Applications

Supervisors: Hans Vatne Hansen (hansvh@ifi.uio.no)

Description:

We surround ourselves with processing capable devices like laptops, smart phones and netbooks all the time. In this assignment, you will develop a mobile application which users can take with them when they move from place to place, and machine to machine.

In the example above, a user initiates a video conference on a desktop computer and seamlessly transfers the conversation onto a smart phone when physically moving away from the computer where the conference was started. When in the vicinity of another trusted computer, the user can choose to transfer the conversion yet again. It is this kind of real-time application where migration really shows its flexibility and advantage, but other applications can benefit from mobility too.

Tasks:

1. **Develop an application** that can benefit from migration. It can be any kind of application that you want. It has to be written in Java and packed into a .jar file for portability reasons.

2. Extend the application to save its state in a .TRAMP file. .TRAMP is an unspecified file format that you may store anything relevant in. (Text or binary, you decide...) In a Solitaire application, the state may consist of the cards and where they are played/placed. In a music

player, the state may be which song is playing and how far it has played. You application should be able to **save some state and read it back to continue execution where it left of**.

3. Optimize the saving and importing of application state. How often should the state be saved? In a real-time application, you need to save it more often than in a slow paced gaming application like Solitaire. Should state be saved at timed intervals or when specified events happens? How can the import be as fast as possible, preferably faster than a human can notice? **Write down your ideas and reasoning.**

4. If you have more time, extend the application with more functionality and state. Make the application better, faster and more impressive! **Do measurements**, i.e. of migration time and responsiveness.

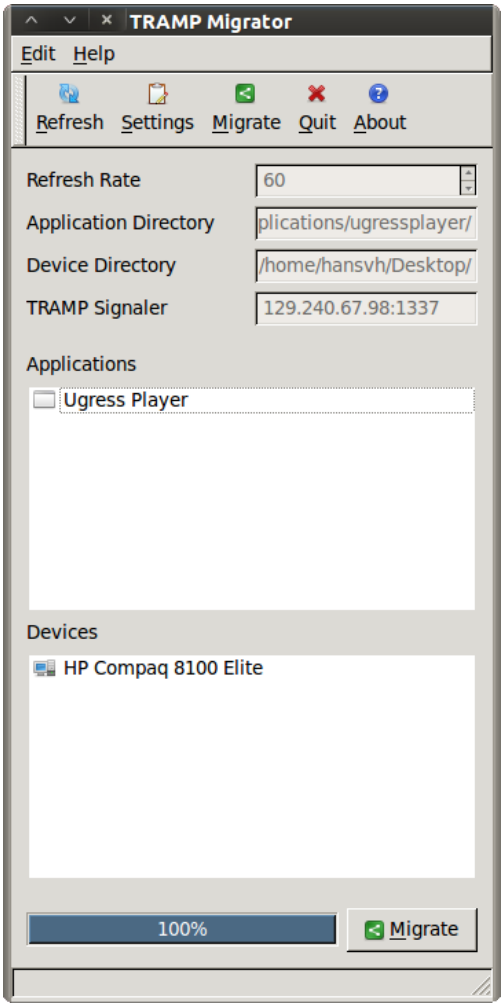
If your group is able to do 1) and 2) that is good. For the best grade, some work needs to be done on 3) and 4) as well. And remember that your written part counts the most!

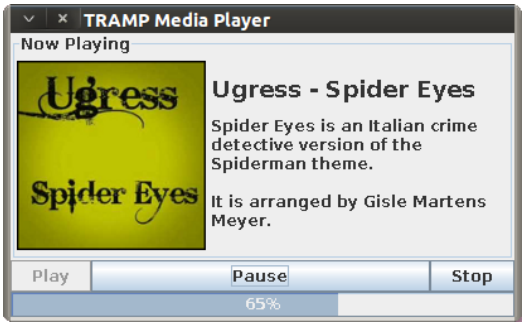
The Platform

Our platform for mobile applications is a research project still in development and you can not expect it to work 100% It consists of two parts, where you will be working on part two. Remember to ask if you are stuck!

1. The TRAMP Migrator, which is the system core responsible for starting applications, stopping them, signalling, moving applications between devices etc. We have chosen to implement our prototype in C++/Qt. This gives us the opportunity the compile it for Microsoft Windows, GNU/Linux and Mac OS. It is both a client and a server.

2. The mobile applications, written in Java. Our first mobile application was a music player, and you can look at it and its source code to see how we are saving state and importing it. Please note, however, that this is not well optimized, and serves only as a proof-of-concept.





Above is a screenshot of our first mobile application. It is rather simple, and can only play one song. Luckily, the song is pretty good. (: When migrated, the player continues to play the song where it left off on the source node.

On the left, the system core is visualized. It shows the running mobile applications and the "mobility-aware" computers in the vicinity. (The latter part is hard coded for now.)

When an application and a device is chosen, a user can click migrate to move a running application and continue execution elsewhere.

Read more about our project at `http://tramp-project.org/`

Required competences:

- Language: Java (Some Qt/C++)
- Tools: Any

References:

[1] Dejan S. Milojivcic et. al., Process migration

[2] George H. Forman AND John Zahorjan, The Challenges of Mobile Computing

Optimizing bandwidth usage in DENS (Distributed Event Notification Service) at a variable publication rate

Supervisors: Piotr Kamisiński (piotr@ifi.uio.no)

Description:

Event notification in MANETs may be a challenging task due to long-lasting network partitions and data loss. To address this problem, we have designed and implemented DENS [1,2], a distributed event notification middleware meant to be run on every node in the MANET. DENS adopts the publish/subscribe paradigm and uses source filtering to limit the number of event notifications (in literature also called publications) sent out by publishers. Subscriptions and notifications are delivered to destinations via special entities called mediators. In order to take advantage of opportunistic contact with other nodes in the MANET, neighbouring mediators periodically replicate with each other the subscriptions and notifications they have acquired. However, this happens at the expense of increased network traffic. DENS already provides mechanisms which can be used to decrease this traffic in specific cases. For instance, in areas with high density of nodes and low relative speed the occurrence of network partitions is unlikely, and therefore replication can be disabled altogether.

Tasks:

In this assignment, we would like to modify the existing periodic replication mechanism, so that it monitors publication rates of publishers, and based on them decide the frequency at which replication should be carried out. The following tasks should be accomplished:

1. Understanding the available mechanisms of replication in DENS.
2. Implementing the necessary modifications.
3. With some help from the supervisors, the students should design a test scenario for the modified replication mechanism, demonstrating its gain over a simple epidemic routing scheme (already built into DENS). Tests (emulations) are to be performed with the NEMAN [3] emulator. The

experiments can be run using the existing test framework (bash and perl scripts around NEMAN and DENS). The following is measured by default through the course of experiments: notification delivery ratio, traffic in the number of DENS packets, traffic in bytes, notification delivery delay.

4. Run emulations of DENS in the created scenario with the new replication mechanism enabled.

5. Run emulations in the scenario with DENS performing simple epidemic routing.

6. Compare and discuss the acquired results. If there is something unexpected, try to explain what could have caused it.

Required competences: Linux, C programming, bash scripting, IP networks, publish/subscribe systems.

A workshop introducing to DENS and NEMAN will be held prior to the start of assignments.

References:

[1] K. S. Skjelsvik, P. Kamisiński, S. Mondet, V. Goebel and T. Plagemann, "DENS: Evaluation of a Distributed Event Notification System for Mobile and Disruptive Environments," to be submitted (printed copy available from the supervisors)

[2] K. S. Skjelsvik, V. Goebel, and T. Plagemann, "Distributed event notification for mobile ad hoc networks," IEEE Distributed Systems Online, vol. 5, no. 8, Aug 2004.

[3] M. Pužar and T. Plagemann, "Neman: A network emulator for mobile ad-hoc networks," in Proc. 8th International Conference on Telecommunications (ConTEL'05), Zagreb, Croatia, Jun 2005, pp. 155–161.

On the evolution of communities in MANETS

Supervisors: Ovidiu Valentin Drugan (ovidiu@ifi.uio.no)

Description:

Identifying the clusters in Mobile Ad Hoc Networks (MANETs) is a challenge, especially because of the network dynamics (i.e., connected nodes, connections between nodes). Community detection methods, the such as the one proposed by Newman and Grivan in 2004 (Girvan & Newman, 2004), can identify the clusters of the network without imposing constraints such as number of clusters or number of nodes in a cluster. Tracking the evolution of the communities over time is an interesting problem and it is important to investigate the lifetime of a community, such as the analysis presented by Green, Doyle and Cunningham in 2010 (Green, Doyle, Cunningham, 2010).

Task:

Design and implement a mechanism to extract topology information from the routing protocol form OLSR (OLSR, 2010) of a node at equal time intervals during a simulation in ns3 (ns-3, 2010).

1. Extract topology information at different time intervals
2. Use a community detection algorithm, for example the Newman and Grivan, algorithm to create to create communities (you can use any graph manipulation library, e.g. igraph (igraph, 2010), or jung (jung, 2010))
3. Perform measurements in static and mobile networks
4. Use different settings for the routing protocol timeouts
5. Build the mechanisms to perform the analysis of the evolution of the communities.

Required competences: Java, C or Python is required and some experience in using Linux is of advantage.

References:

[1] Girvan, M., & Newman, M. E. (2004). Finding and evaluating community structure in networks. Physical Review E , 69 (2).

[2] Greene, D.; Doyle, D.; Cunningham, P.; , "Tracking the Evolution of Communities in Dynamic Social Networks," Advances in Social Networks Analysis and Mining (ASONAM), 2010 International Conference on , vol., no., pp.176-183, 9-11 Aug. 2010

[3] igraph. (2010). From igraph library: `http://igraph.sourceforge.net`

[4] jung. (2010). From jung libray `http://jung.sourceforge.net/`

[5] ns-3. (2010). From ns-3: `http://www.nsnam.org`

[6] OLSR. (2010). www.olsr.org. From `http://www.olsr.org`

Image retargeting for handheld mobile devices

Supervisors: Benjamin Guthier (guthier@informatik.uni-mannheim.de)

Description:

In most cases, the original resolution of digital photos is much higher than the screen resolution of a mobile devices. Sending all image data wastes a lot of bandwidth, and causes additional processing load on the mobile device for image decompression and resizing. In addition, the visual quality caused by scaling is not always satisfactory, especially if anisotropic scaling is used which applies different scale factors for the width and height of an image.

A better solution would be to adapt images on a server and only transmit the small image to the mobile device. This solution would also allow to use advanced (and more complex) algorithms for image retargeting. In this assignment, the students should use a mobile phone to connect to a web page. The display resolution and the retargeting technique are sent to the web server. The server processes this information, adapts the image, and sends it back. Seam carving [1-3] is a novel image retargeting algorithm that provides good results in many cases. The students should implement the seam carving algorithm and compare the quality of adapted images to scaling, cropping, and letter boxing.

Tasks:

1. Get an overview of novel image retargeting techniques [4].
2. Implement three simple image retargeting techniques (e. g., scaling, cropping, letter boxing) and the advanced seam carving technique [1-3]. You can use libjpeg [5] to open and save JPEG images.
3. Use a mobile phone (you can also simulate it in a web browser) to connect to a web page. Communicate the target resolution and the adaptation technique to the web server. Apply the selected image retargeting technique, send the new file to the mobile phone, and visualize the image.
4. Execute experiments and analyse the quality and computational effort of the implemented retargeting algorithms.

5. Write a 10 page report (IEEE style) which discusses the implemented retargeting algorithms.

Required competences: Basic programming skills in JAVA, C, or C++

References:

[1] Shai Avidan and Ariel Shamir. 2007. Seam carving for content-aware image resizing. ACM Trans. Graph. 26, 3, Article 10 (July 2007).
[2] `http://www.youtube.com/watch?v=vIFCV2spKtg`
[3] Ariel Shamir and Shai Avidan. 2009. Seam carving for media retargeting. Commun. ACM 52, 1 (January 2009), 77-85.
[4] Michael Rubinstein, Diego Gutierrez, Olga Sorkine, and Ariel Shamir. 2010. A comparative study of image retargeting. In ACM SIGGRAPH Asia 2010 papers (SIGGRAPH ASIA '10).
[5] LibJPEG, `http://www.ijg.org/`

Fountain codes for highly unreliable networks

Supervisors: Benjamin Guthier (guthier@informatik.uni-mannheim.de)

Description:

Fountain codes are a fairly recent discovery in the field of source coding with high relevance for unreliable networks but also for high latency scenarios. A very brief description of the idea:

Traditional data transmission (e.g., FTP) splits a file into small so-called data chunks and sends each chunk as a separate packet until it has been fully transmitted. A missing or erroneous data chunk has to be replaced by another copy. In case of a large number of receivers like in broadcast or multicast scenarios, many chunks might have to be retransmitted. The process ends no sooner than everyone has got the full set of chunks. If the link is of low quality and the delay is very long or if no feedback channel is available at all, providing all receivers with the full set of chunks to reconstruct the data might be challenging or completely infeasible. This is where data fountains come in.

The data fountain sends a potentially endless amount of packets which are pairwise different with a high probability. In other words, you will hardly ever see a packet twice. Let us assume that a file was split into 10 data-chunks. With the traditional FTP-like data transfer, a receiver needs to get chunk #1 to #10 in order to reconstruct the file. Using a data fountain, we can pick any 10 packets from the endless stream of packets and this random choice of 10 packets will reconstruct our file in most cases. In rare cases we might need a small number of additional packets. The unusual property, however, is that any missing or erroneous chunk can be replaced by any other chunk (with a very high likelihood). In contrast, in the case of FTP, a missing chunk #5 can only replaced by another copy of chunk #5. In the case of data fountains, a single additional packet will replace missing packets at the receivers' side, no matter were they occurred. Intuitively speaking, an additional packet is useful for closing any gap at the receivers side. So a very low number of additional packets might already be enough to satisfy all receivers. This is relevant e.g., for file transfer in peer-to-peer networks with many receivers.

Task:

1. Get an overview over existing fountain code algorithms.
2. Implement a sender that splits a data file (e.g., an image) into chunks and encodes them with fountain codes.
3. Implement a receiver that receives the chunks, does the necessary decoding and reconstructs the original data.
4. Connect sender and multiple receivers. The sender should simulate random packet loss, so that each receiver gets a different set of chunks. Experiment with varying loss rates.
5. Visualize the intermediate states of data reconstruction on the receiver side.

Required competences: Basic programming skills in Java or C/C++

Analysing the Impact of Internet Worm Attacks Using a Simulation Environment

Area: Network resilience
Supervisors: Andreas Mauthe (a.mauthe@lancaster.ac.uk), Alberto Schaeffer-Filho (asf@comp.lancs.ac.uk)
Description:

Networks must be resilient to challenges such as malicious attacks or network overload and adapt their operation in an autonomous manner. Resilience is the ability of the network to maintain acceptable levels of operation in the face of challenges to its operation. However, it is difficult to evaluate resilience strategies that involve the interplay between a number of detection and remediation mechanisms that must be activated on demand according to events observed in the network (as opposed to hardcoded protocols). For this reason, we have developed a policy-based resilience simulator [1] based on the integration between the OMNeT++ [2] network simulator and the Ponder2 [3] policy framework. The policy-based resilience simulator allows the use of real Ponder2 policies to control the operation of instrumented, policy-enabled components running within the simulation. It is then possible to understand how real policies affect the operation of resilience mechanisms running within the simulation environment, and then evaluate resilience strategies before they are implemented and deployed in the network infrastructure.

This project consists in implementing an attack profile simulating one or more types of worm attacks, and implementing a module for its detection [4], [5]. Other resilience components already implemented in our existing resilience simulation platform can also be used (e.g. link monitor, rate limiter). The student will analyse the impact on the network of varying forms of worm propagations and evaluate the policies for remediating the attack in the simulation, possibly in a number of different network topologies. Download and installation instructions for the toolset can be found at: `https://forge.comp.lancs.ac.uk/hosted/resilience/policy-resilience-simulator`.

Tasks:

1. Understand OMNeT++ topologies, traffic profiles and its general configuration

2. Study the different forms of Internet worm attacks and propagations

3. Write an attack profile simulating one or more types of worm attacks at varying intensities

4. Implement a detection module as an instrumented component in our simulation platform to identify the attack (e.g. using payload inspection, traffic features, etc) and define its management interface (inputs/outputs)

5. Configure policies to successfully limit the effects of the worm attack in the simulation (using already implemented policy-enabled modules for link monitoring and rate limiting)

Required competences: C++, OMNeT++

References:

[1] A. Schaeffer-Filho, P. Smith, and A. Mauthe. Policy-driven network simulation: a resilience case study. In 26th ACM Symposium on Applied Computing (SAC 2011), Taichung, Taiwan, March 2011. ACM.
[2] OMNeT++ Website. OMNeT++. Available at: http://www.omnetpp.org/. Accessed February 2011.
[3] Ponder2 Website. Ponder2. Available at: http://ponder2.net. Accessed February 2011.
[4] T. Gamer and C. P. Mayer. Large-scale evaluation of distributed attack detection. In Simutools '09, pages 1–8, Rome, Italy, March 2009. ICST.
[5] T. Gamer, C. P. Mayer, and M. Zitterbart. Distack: a framework for anomaly-based large-scale attack detection. In SECURWARE '08: Proceedings of the 2008 Second International Conference on Emerging Security Information, Systems and Technologies, pages 34–40, Cap Esterel, France, August 2008. IEEE Computer Society.

Implementation and Evaluation of Internet Traffic Classification Algorithms

Area: Network resilience
Supervisors: Andreas Mauthe (a.mauthe@lancaster.ac.uk), Alberto Schaeffer-Filho (asf@comp.lancs.ac.uk)
Description:

Networks must be resilient to challenges such as malicious attacks or network overload and adapt their operation in an autonomous manner. In order to evaluate resilience strategies that involve a number of detection and remediation mechanisms, we have developed a policy-based resilience simulator [1]. This is based on the integration between the OMNeT++ [2] network simulator and the Ponder2 [3] policy framework, which allows the use of real Ponder2 policies to control the operation of instrumented, policy-enabled components running within the simulation. The simulation platform allows the evaluation of resilience strategies before they are implemented and deployed in the network infrastructure. One of the key problems related to a resilience strategy is to reliably discriminate anomalies in the network, for example, operational overload due to a flash crowd or due to malicious attacks, and then apply adequate counter-measures. Algorithms for network traffic classification allow the characterisation of properties of the network traffic, and these apply techniques from several research domains, ranging from statistical analysis to machine learning.

This project consists in looking into existing algorithms and reference implementations for internet traffic classification, including those found in the well-known Weka project [4], and porting one or more of those algorithms to our OMNeT++ based simulation platform. In the past, researchers have ported the implementation of detection algorithms to the OMNeT++ platform [5], [6]; in this assignment, the student will port classification algorithms to the OMNeT++ platform. The student will then evaluate the performance of the algorithms under different simulation scenarios. The output of this project will provide practical insights about which algorithms perform better in different situations. Download and installation instructions for the resilience simulator toolset can be found at: `https://forge.comp.lancs.ac.uk/hosted/resilience/policy-resilience-simulator`.

Tasks:

1. Understand OMNeT++ topologies, traffic profiles and its general configuration

2. Study the different techniques for traffic classification and the main algorithms within each area

3. Design a traffic classification module as an instrumented component in our simulation platform, implement one or more algorithms for traffic classification and define their management interfaces (inputs/outputs)

4. Evaluate the performance and accuracy of the classification algorithm(s) using a DDoS attack scenario (which is already implemented)

Required competences: C++, OMNeT++

References:

[1] A. Schaeffer-Filho, P. Smith, and A. Mauthe. Policy-driven network simulation: a resilience case study. In 26th ACM Symposium on Applied Computing (SAC 2011), Taichung, Taiwan, March 2011. ACM.
[2] OMNeT++ Website. OMNeT++. Available at: http://www.omnetpp.org/. Accessed February 2011.
[3] Ponder2 Website. Ponder2. Available at: http://ponder2.net. Accessed February 2011.
[4] Machine Learning Group at University of Wiakato. Weka. Available at: http://www.cs.waikato.ac.nz/ml/weka. Accessed February 2011.
[5] T. Gamer and C. P. Mayer. Large-scale evaluation of distributed attack detection. In Simutools '09, pages 1–8, Rome, Italy, March 2009. ICST.
[6] T. Gamer, C. P. Mayer, and M. Zitterbart. Distack: a framework for anomaly-based large-scale attack detection. In SECURWARE '08: Proceedings of the 2008 Second International Conference on Emerging Security Information, Systems and Technologies, pages 34–40, Cap Esterel, France, August 2008. IEEE Computer Society.