Implementation of Octopus-based Key Agreement Protocols

Maria Striki, Kyriakos Manousakis, John Baras

In this demo, the real-time implementation of a set of group communication key agreement protocols for mobile ad hoc networks (MANETs) is demonstrated. These protocols are extensions of the hybrid Octopus key agreement scheme, originally designed for wire-line networks. The protocols we are demonstrating are: the Octopus 2^d (O), and the modified Octopus 2^d with Tree (MOT). The demonstration will include both the basic functionality of the previously mentioned protocols for the establishment of a multicast group key (including the multicast group formation) and also the response of the protocols in cases of dynamic membership changes in the multicast group (detection of group change and new key establishment). The presentation of the protocols will be supported from a GUI and a user level multicast application which will encrypt, transmit, receive and decrypt multicast data (i.e. pictures). The application will encrypt and decrypt data by utilizing the established multicast group keys from the application of each one of the demonstrated protocols. We will also demonstrate the real-time implementation of an Entity Authentication scheme integrated with these key management protocols.

The Future Battlefield Networks will involve thousands of heterogeneous nodes, arranged in a hierarchy, operating under dynamically changing mission needs and connectivity: *Hierarchical MANETs.* **Group Communications** are the pervasive mode of communications in MANETs and *sensor* networks. **Cryptographic keys** and their **efficient** (in terms of *Bandwidth*, *Energy*, *Storage* and *Computation*) **management** are the **critical** instruments for maintaining **security** in these battlefield networks. The design of efficient key management schemes for the resource-constrained environment of MANETs is of paramount importance, since the performance of the corresponding key management functions imposes an upper limit on the efficiency and scalability of the whole secure group communication system.

Our Solution: We contribute towards secure, scalable and efficient group communication for MANETs, with the **design of two new protocols** (GDH.2-*based modified Octopus* MO and TGDH-*based modified Octopus with Tree* MOT), and **the adaptation of the** *original Octopus* **(O)** to MANETs.

Operation: Hierarchy is supported through the partition of a large key agreement (KA) group to 2^d subgroups of smaller size. Initially, each subgroup agrees on its own subgroup key, and elects a subgroup leader to handle KA locally within its own subgroup. In MO and MOT, each subgroup executes a non-centralized (GDH.2) or a hybrid (TGDH) subgroup key generation protocol, as opposed to the original (O), where each subgroup executes a centralized scheme that resembles GKMP. Then, the subgroup leaders alone, representing their subgroups, interact among themselves and use the previously generated subgroup keys to agree on a global group key via another KA protocol, the Hypercube. Finally, the subgroup leaders distribute securely the group key to their subgroup.

Motivation: The following features of Octopus schemes have motivated our interest to extend them: *a*) their **hierarchical framework** through which they can tolerate network dynamics and be more scalable, *b*) a **subgroup leader** handling a relatively **small subgroup** is a feasible option in MANETs, as opposed to a single leader in centralized schemes handling a very large group, *c*) **disruptions** are **handled locally** at first within the subgroup, and then reflected to the whole group via low cost re-keying over Hypercube.

Goals Achieved: Superior performance (especially with MOT compared to the original Octopus) (schemes also compared to centralized OFT): *Robustness, Communication & Computational Overhead Reduction*

Highlights: The protocols were implemented on Linux (kernel-2.4.18), and the OpenSSL library has been used for the symmetric key encryption (DES method) whenever required. A multicast application that transmits encrypted data with the established group key is additionally implemented.

What is demonstrated:

- Participating Nodes to the Secure Group Establish a Common Session Key
- Dynamic Addition and Deletion of a Member Node to the Secure Group
- Nodes that no longer belong to the secure group cannot receive the transmitted encrypted data

TestBed Configuration:

- The leaders are the 4 laptop P3 machines running LINUX RedHat 8.0.
- The Subgroups are dynamically generated.
- The leaders are sensing the subgroup changes and re-keying is triggered.