Towards Benchmarking of P2P Technologies from a SCADA Systems Protection Perspective

Abdelmajid Khelil, Sebastian Jeckel, Daniel Germanus, and Neeraj Suri[*]

Technische Universität Darmstadt,
Hochschulstr. 10, 64289 Darmstadt, Germany
Tel.: +49 6151 16{3414—3711—5321—3513}; Fax.: +49 6151 16 4310
{khelil,jeckel,germanus,suri}@cs.tu-darmstadt.de

Abstract. Supervisory Control and Data Acquisition (SCADA) systems are used to control and monitor critical processes. Modern SCADA systems are increasingly built with off-the-shelf components simplifying their integration into existing networks. The benefits of increased flexibility and reduced costs are accompanied by newly introduced challenges regarding SCADA security/dependability. Peer-to-Peer (P2P) technologies allow for the construction of self-organizing, dependable and large-scale overlays on top of existing physical networks.

In this paper, we build the base for using P2P to enhance the resilience of deployed SCADA systems. To this end, we provide a general analysis of both domains and their compatibility. In addition, we refine the existing classifications of P2P technologies w.r.t. the needs and capabilities of SCADA systems. Consequently, we identify core P2P-based protection mechanisms for SCADA systems, based on data and path replication. Our main results are generic guidelines for the exploitation of P2P technologies to enhance the SCADA resilience.

Keywords: SCADA, Critical Infrastructure Protection, P2P, Dependability, Security.

1 Introduction

For life in modern-day societies the dependability of Critical Infrastructures (CI), e.g., power grid or water supply, is of essential character. Supervisory Control and Data Acquisition (SCADA) systems are embedded in these CI for the purpose of monitoring and controlling them. While the first SCADA systems were built using proprietary standards and dedicated hardware in closed architectures, the trend is towards more flexible systems and open protocols like the Internet Protocol (IP). IP-enabled SCADA components allow usage of commercial off-the-shelf (COTS) products and integration into existing network structures, e.g., corporate LAN or WAN like the Internet, thus saving costs of specialized hard-/software and allowing faster adaption to changing requirements. At the same time, this technological shift towards a networked system, eventually

[*] Research supported in part by EU INSPIRE, CASED (www.cased.de), and EU CoMiFin.

even connected to the Internet, introduces new threats and vulnerabilities to SCADA systems and since the disputed concept security through obscurity is no longer applicable, previously unnoticed or ignored security issues might now be exposed. To handle these security challenges, techniques from conventional networked systems can be transferred to the SCADA domain.

Unlike the classical client/server concept, where roles are strictly separated, in Peer-to-Peer (P2P) networks every node/peer is a client as well as a server. While at first mainly used for file-sharing, the active research attention P2P received led to several new application areas. There is a clear trend away from client/server towards P2P-based decentralized, self-organizing and fault-tolerant architecture. Similarly, the migration from the deterministic and client/server-like SCADA architectures towards P2P-like architectures is desired [1,2,3] in order to increase the SCADA resilience. However, such paradigm switch is challenging for operational, heterogeneous and difficult-to-replace SCADA systems. P2P overlays, through their inherent data and path replication mechanisms, allow to break the determinism of already deployed SCADA systems and consequently provide for a promising SCADA protection approach. The main contributions of this paper consist in providing generic guidelines for using P2P technologies to protect operational SCADA systems. In particular, we show how an appropriate P2P technique can be selected for a given SCADA system. This mapping is based on the SCADA system's properties, requirements and the specific application.

The rest of the paper is organized as follows. Section 2 gives an overview of SCADA systems, highlighting their characteristics and special needs. In Section 3, we briefly describe the most prominent P2P techniques and their underlying concepts and design motivations. Consequently, we provide a novel fine-tuned classification of P2P techniques according to SCADA-relevant properties. Section 4 details the requirements, prerequisites and architecture for P2P-enabling while discussing the potential usage scenarios of P2P to enhance the resilience of SCADA systems. In Section 5, we describe how one should proceed to select an appropriate P2P technique for a specific SCADA system. Section 6 presents conclusions and directions of future work.

2 SCADA Systems Overview

The purpose of SCADA systems is to allow remote human supervision of critical processes found in infrastructures, industrial sites or other facilities. The two main tasks of SCADA systems are (a) to collect process data and present it to an operator, and (b) to forward operator commands to the process where they get executed. In the following we will give an overview of how SCADA systems are built and how they operate.

SCADA systems are organized hierarchically. At the lowest level, sensors and actuators are embedded directly within the industrial processes. At the next level, Remote Terminal Units (RTU) are connected to possibly multiple sensors or actuators. A wide range of components can be used as RTU, such as ordinary personal computers, Programmable Logic Controller (PLC), and small-sized/custom devices. RTUs send their collected data via LAN or WAN to a

Master Terminal Unit (MTU). An MTU gathers accumulated sensor data from RTU, monitors the system and sends commands to actuators to adjust the system behavior. MTUs vary in size and can house (1) Human Machine Interface (HMI) stations, which visualize the system state for human operators, (2) database servers to store received information for record-keeping and analysis, (3) data acquisition nodes, which receive incoming data and forward it for further processing, and (4) data processing nodes to compute calculations on the received data and execute automated control loops. It is also possible for an MTU to consist of a single computer with only HMI software and a human operator. For large scale SCADA systems spanning over a wide geographical area, the hierarchy is extended with additional layers of supervisory stations.

The emerging networked topologies are highly flexible but also highly heterogeneous. Networked SCADA systems may be interconnected across whole nations and beyond, just as the corresponding infrastructures like power grids are. [4] gives an example for such a large scale system to monitor a power grid in the US, consisting of 270 utility stations at different locations with up to 50000 sensors at each of them. Fig. 1 shows an example topology for a networked SCADA system.

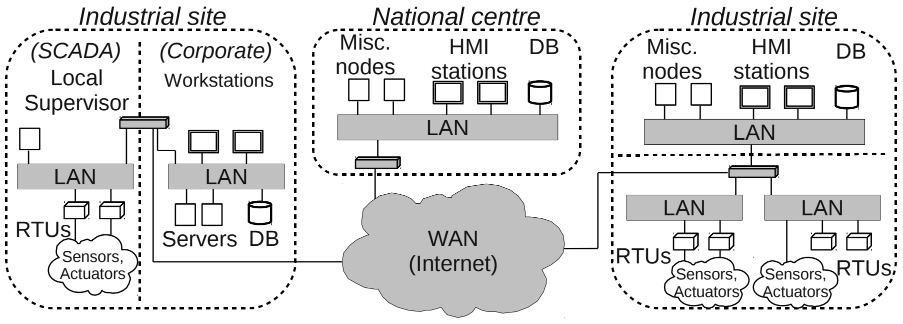

Fig. 1. Networked SCADA system

Data Flows and Timeliness Requirements: Three types of control loops can be identified. (i) Safety-critical control loops are realized at RTU level and normally implement emergency protocols to shut down or throttle the process if communication with the supervisor is interrupted or a critical incident is imminent. The timeliness constraints for such control loops are in the millisecond-to-second range to allow very fast responses to potential safety threats. (ii) Operation-critical control loops are necessary for the process to perform its tasks properly. They may involve several entities at supervisory level, including human operators. Since one RTU usually is just one among many, these control loops handle coordination backed by system-level intelligence and aggregated information to realize complex tasks that involve multiple entities. The timeliness requirements are less strict here and delay up to several seconds or even minutes might be acceptable depending on the concrete scenario. However, to enable remote supervision of a process in real-time, low latency for these control loops is desirable.

(iii) Non-critical control loops are used for process optimization, record-keeping, higher-level coordination, etc. Their failures can be compensated and do not critically affect the system. Summarizing, there are two main, unidirectional data flows in a SCADA hierarchy: (1) Sensor data, upwards from sensors to higher levels, and (2) control commands, downwards from higher levels to actuators. In addition, multiple supervisory stations may exchange their data.

The Main SCADA Perturbations: Based on [5] the perturbations identified for digital, packet-based messaging of SCADA are delay, jitter, transient loss, permanent loss, eavesdropping and injection. Delayed messages are prevented from arriving at their destination in time. For messages from sensors this could mean that the operator is informed of dangerous situations too late. Jitter is a form of delay, where messages arrive out of order so they fail their intended purpose. Similarly, transient or permanent message loss may disturb the proper operation of the SCADA system. Unlike these accidental perturbations, eavesdropping and message injection are deliberately conducted by an attacker. Usually, eavesdropping serves the purpose of collecting data about the system used to prepare further attacks. With message injection an attacker can send his own commands to actuators or report fake sensor data to operators, either to hide the real process state or to maneuver them into taking inappropriate actions.

3 SCADA-Driven Classification of P2P Technologies

After briefly surveying the existing P2P technologies, we present a new classification for the selection of an appropriate P2P technology to protect SCADA systems in the presence of faults.

3.1 Overview of P2P

Generally, an overlay network - or short just overlay - is a virtual network constructed on top of another physical one, which is called underlay accordingly. Comprehensive surveys of existing P2P technologies can be found in [6] and [7].

First-generation Systems - Unstructured Overlays: The first applications that popularized the term P2P and its underlying technologies were mainly used to share digital music or other small media files for a large number of Internet users. The traditional client/server architecture shows both a poor scalability and fault-tolerance since it centralizes the storage, indexing, search and transfer of data. Instead of storing all the data, the first P2P technique Napster uses a server only to maintain an index of all shared objects and the corresponding users and to provide for an efficient search. The file storage and transfers only involve the users. Gnutella [8] provides a fully decentralized P2P architecture. This is achieved by decentralizing the storage, indexing, search and transfer of data, which is now completed by users/peers without relying on any centralized entity. Gnutella mainly uses flooding for topology and data management. When new peers join a Gnutella overlay they are more likely to connect to an already well connected node (the integration of new peers follows a power law). Consequently, the performance and fault-tolerance of the overlay depends on these well-connected node. Performance studies pointed out Gnutella's poor performance. Gia [9] improved Gnutella performance through a designing a flow control mechanism, using random walk instead of flooding and adapting the topology to the capacity of peers. FastTrack [10] improved Gnutella's design and made it scalable with acceptable performance. It did so by exploiting the invariants that peers are highly heterogeneous w.r.t. to their capabilities and that the probability for a peer to leave the network is inversely proportional to its current up-time. Where Gnutella treated all peers equal, FastTrack promotes ordinary peers that are able and willing to take over more responsibilities to so called super peers.

Second-generation Systems - Structured Overlays: The decentralized P2P systems introduced so far have problems to locate data efficiently and correctly at the same time, especially for systems with many peers. Compared to centralized systems, they lack a global index that contains the location of all data objects existing within the network. Trying to maintain such a complete index for every peer individually does not scale because each one would need to keep track of every object in the network, resulting in huge coordination complexity. The P2P systems described next use the concept of a Distributed Hash Table (DHT) to replace the keyword-based searching found in first-generation systems. For a DHT, the hash buckets are distributed among the peers in an overlay network. So to look up the value associated with a given key, the peer responsible for the respective bucket has to be found first. Therefore, the supported operations of a DHT are reduced to lookup(k) with changed semantics so that it returns the peer responsible for the given key k. To retrieve, store or delete a value this peer has then to be contacted directly. The terms routing time and look-up time can be used interchangeably, as the look-up of a key simply means routing towards it. The challenge of implementing a DHT is to realize overlay routing for a fully decentralized topology, so that starting from an arbitrary peer, the one responsible for any given key can be found in a fast and efficient way.

There are many different DHT implementations and the most prominent representatives are Chord [11], CAN [12], Tapestry [13], Kademlia [14], Pastry [15], Koorde [16] and Kelips [17].

P2P Applications: Generally, an overlay can extend the underlay's features or modify its behavior, for example by adding an additional layer of security like VPN do, without breaking compatibility or making changes to existing protocols. This makes overlays especially attractive for use over networks like the Internet, where modification of currently used protocols is not an easy option. P2P systems use application-layer overlay networks to realize decentralized organization such as efficient multicast/streaming [18], distributed storage [19] or event notification [20]. While the last two could also be put to use in SCADA systems, the primary interests are overlay features and applications that increase resilience. [21] uses a structured overlay to overcome link failures by replicating data at other peers. In [3] a concrete implementation of overlay networks for power grids is presented. These approaches are considered as a starting point to explore additional P2P applications or techniques that could be used in a general SCADA system context.

3.2 P2P Classification

Because the conventional classification of decentralized system (structured vs. unstructured) only relates to their basic underlying principles and does not cover any specific details, in this section fully distributed P2P technologies will be classified w.r.t. to the aspects relevant for SCADA protection.

Benchmarking Criteria: The first step necessary to effectively compare and benchmark P2P systems is an analysis of their distinctive properties and features. To capture the relevant properties of P2P systems, the following indicators are commonly used: Topology of the overlay, average length of paths, routing overhead, maintenance overhead, average peer degree, overlay dynamics, overlay determinism, routing latency, state size on peers, self-optimization, locality, load balancing, heterogeneity of peer roles, overlay robustness and security level. Evaluation and benchmarking criteria should consider these indicators. With the SCADA applications in mind, the focus is on the following benchmarking criteria for given reasons: (1) Routing latency, due to stringent SCADA timeliness requirements, (2) minimum requirements, because of limited capabilities of legacy SCADA components, (3) communication overhead, since P2P traffic should not block SCADA communication, and (4) robustness and security, as these are the aspects that should be improved for SCADA systems.

Routing Latency: The overlay capabilities of structured and unstructured systems differ as the former can use directed routing while the latter have to search blindly, for example by flooding. This has a significant effect on the upper bound of routing steps. For structured systems with N peers, this is $O(\log N)$ or better, depending on the implementation. For unstructured systems it is $O(N)$. Albeit this worst case is highly unlikely, especially when using optimized architectures like [9], it generally shows that for large systems paths are not theoretically bounded in length and may become very long (Fig. 2 left). Short paths are crucial for low latency as each additional step not only brings obvious additional delay but also raises the chance to encounter a low quality link. However, there is no guarantee that these may not be found on short paths as well. That is why to truly minimize routing latency self-optimizing and/or locality-aware techniques have to be used. Finding short paths also depends on the node degree: The higher the number of neighbors, the higher is the chance to find one close to the destination.

Minimum Requirements: The amount of required memory mainly depends on the node degree. All introduced structured networks have fixed upper bounds that are supposed to be reached because of overlay-specific invariants required for efficient routing. This allows little flexibility for individual peers w.r.t. heterogeneous capabilities, so the weakest peers in the network actually dictate the maximum possible state size for all others (Fig. 2 right). Unstructured networks have better support for heterogeneity because in a random overlay it doesn't matter if some peers are only able to manage a few connections as long as the average node degree is high enough to keep the graph connected. For peers with very low capabilities, hybrid topologies can relief them of any P2P-specific requirements so they are just in a client/server relationship with another peer.

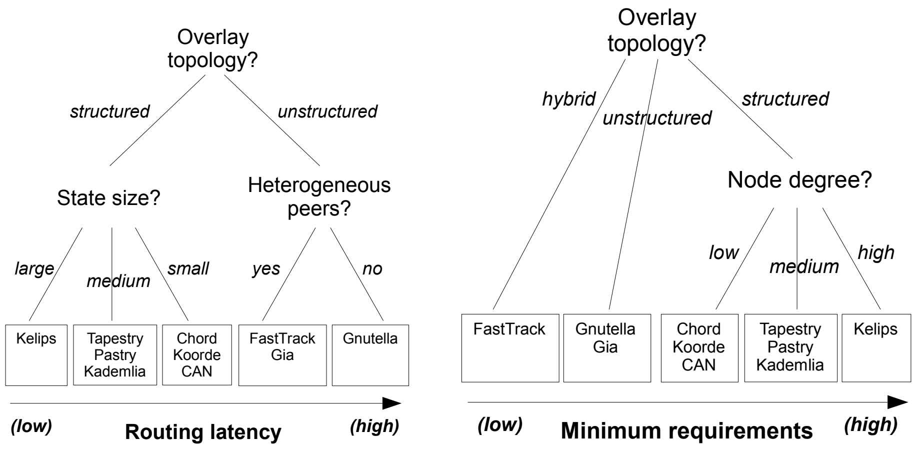

Fig. 2. Routing latency and minimum requirements

Communication Overhead: What is included here is the overhead for routing and maintenance (Fig. 3). The former varies with the supported overlay operations since flooded messages spread exponentially resulting in excessive amounts of unnecessary communication when only addressing a single target. Structured systems, on the other hand, usually don't cause additional routing traffic unless it is for redundant messages or concurrent routing to increase reliability or lookup speed. The influences on maintenance overhead are state size and specifically node degree. Connections to neighbors have to be kept up-to-date by checking for disconnected peers, updating meta-data like roundtrip time, etc. Rigid and complexly constructed overlay structures generally result in higher message overhead to integrate new peers or handle membership changes. This implies that unstructured systems have little maintenance overhead, or at least for them it can be reduced to a low level because there is no fear that the overlay structure might degrade. The deployment of P2P should not impact negatively the underlying SCADA application by excessively consuming network bandwidth. To overcome this issue, [22] proposes to model the overlay network and bandwidth availabilities as a minimization problem. The problem is NP-Hard and a heuristic for a distributed algorithm to continuously adapt P2P neighbour relationships was developed. The adaption results in topologies that do not excessively stress peers with low bandwidth capacities. This supports high message availability and low timeliness because messages do not get stuck due to congestion. The overlay adaption process is repeated regularly because bandwidth provision in large-scale networks is subject to variations.

Depending on the implementation, dynamic topologies, for example caused by self-optimization, either have a positive or a negative impact on maintenance. That is because when using routing messages to update the overlay at the same time, no explicit maintenance communication is necessary. On the other hand, if self-optimization causes additional messages this increases the overhead. The

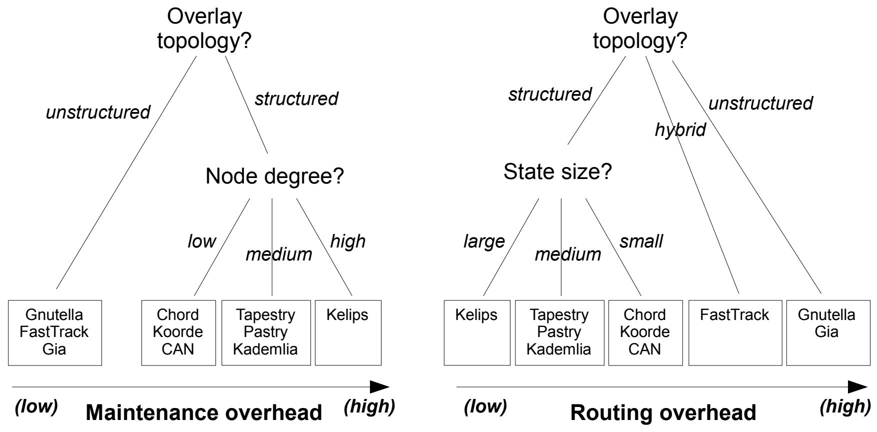

Fig. 3. Communication overhead

same statements in the last paragraph about state size and support for heterogeneity hold here as well.

Robustness and Security: Examining robustness against random failures for unstructured overlays, [23] shows that when comparing pure random graphs to those constructed by a power law, the latter are much more resilient when a very high number of peers (e.g., more than 20%) are dropped. The ability of structured systems to handle randomly failing peers depends on node degree as shown in [16]. With an average degree of $O(\log N)$, a structured system can theoretically recover even if 50% of the peers fail. The situation changes for targeted attacks. Generally, whenever responsibilities are concentrated at a few nodes, for example because of heterogeneous capabilities, these make preferred targets for an attack as their breakdown has great potential to disrupt the overlay. This also holds for power law networks when attacking the nodes with highest degree. Though only having an indirect impact, load balancing also helps to increase overlay robustness as it counteracts failure of nodes due to an overwhelming number of requests. Failures that are locally concentrated, for instance caused by the effects of a disaster, also lead to multiple overlay failures in close proximity to each other. It would be safer regarding overlay stability if these failures were evenly distributed across the overlay. The most important aspect about security is how easy it is for an attacker to make reasonable assumptions about the system. Rigid, deterministically constructed overlays as found in some structured systems allow an attacker who is able to deduce information from intercepted messages to build up knowledge about the system state; it may even be possible to predict behavior purely based on algorithmic details. Dynamic topologies make it hard to learn about the system and consequently complicate a successful attack. Due to the decentralized nature of P2P systems, attackers are required to control either a large fraction of peers in the network to gain control over it, or they are required to introduce peers at specific locations either in the address space or in the topology to enable their malicious intent. Two famous attacks in this context are the Sybil attack [24] and the Eclipse attack [25]. Both attacks require either introduction or hijacking of notable amounts of peers. The following two basic measures may avert these attacks. (1) The peer ID assignment needs to be safe, e.g., physical machines must not be admitted to join the overlay network with multiple IDs at the same time. (2) Topologies should not be open for extension, i.e., contrary to file sharing networks, some serious applications may define a fixed set of nodes that is allowed to participate in the overlay network and thereby excludes newly introduced nodes on behalf of an attacker. The second measure can be realized by using an admission protocol [26].

4 Increasing SCADA Protection with P2P Technology

Now that an overview of both SCADA and P2P technologies has been given, we will attempt to bring them together. This is achieved by discussing general feasibility, technical prerequisites and architecture, and analyzing prospective applications.

4.1 P2P-Enabling: Requirements, Prerequisites and Architecture

Because of the critical nature of security and dependability for SCADA systems, the following basic objectives should be followed when applying P2P techniques: (1) The underlying SCADA system's dependability and in particular QoS[1] should not be degraded and the overall system resilience should be increased, and (2) solutions to newly introduced P2P vulnerabilities must exist and their costs should not outweigh the P2P benefits. The P2P overlays introduced so far are built for use in a large-scale network, namely the Internet, so to qualify as a peer a computer must be connected and implement the Internet Protocol Suite to be able to communicate. The INSPIRE project [27] shows how P2P can be integrated into the SCADA message flow by adding an additional middleware layer, where all P2P-related functions are realized. This middleware should be supported by an underlying operating system for a peer to take a full part as a member of the overlay.

We follow an intrusive, active monitoring approach of SCADA systems, contrary to [1,2], which design a P2P-based passive monitoring architecture for smart micro power grids as a specific type of SCADA system. Working with a fixed and well-known population even allows for a minimal overlay maintenance overhead. As P2P architectures are usually open for new, unknown participants, these systems inherently do not offer any methods for access control. These have to be implemented since SCADA systems are necessarily closed and must only consist of well-known entities.

4.2 Potential P2P Benefits for SCADA

Considering a representative scenario in which an interruption of the SCADA message flow has been detected, e.g., from an RTU to a server in the MTU, we preset

[1] QoS refers to message delivery time, quantifiable by calculating values for mean, maximum and standard deviation.

different possible applications that can be implemented when using a P2P overlay to maintain the SCADA resilience. Naturally, as structured and unstructured architectures differ in functionality, so does the way these applications on top of them are implemented. The two basic methods to exchange data between the otherwise disconnected entities are rerouting and replication. Rerouting means that the RTU (re)sends its messages via overlay to the server, representing a push approach. For replication on the other hand the RTU's data is replicated at alternative locations where the server can pull it from. Though per definition replication has a slightly larger communication overhead since it is done in two steps, ultimately the SCADA application may dictate which method to use. Another distinction is made between direct and indirect rerouting: Either the overlay is used to directly transfer the message itself, or just to find and establish an intact path to the server that can subsequently be used. Which method is more efficient depends on multiple factors: For large message sizes it is better to find a short, high quality path first instead of transmitting large amounts of data over multiple hops with potential delay and/or low throughput; the same goes for frequent message loss, in which case one path can be reused multiple times. For small messages and/or transient loss direct sending is better suited since there is no overhead. For replication, the two basic approaches are to either wait until a failure has been detected and then react, or to proactively replicate all data regardless of failures. Unlike reactive replication, the latter strategy will cause permanent background traffic and require a constant amount of memory. It has the advantage that no coordination between the replicating RTU and the server is necessary and failure detection capabilities are only required on the server side. In the following, we discuss general implementation strategies for rerouting and replication in unstructured and then structured P2P overlays.

Applications for Unstructured P2P Systems. *Rerouting:* As explained earlier, unstructured systems do not support directed routing towards a certain destination; instead the overlay has to be searched randomly, e.g., by using a distributed algorithm based on random walk or flooding. What is different for routing compared to object search is that ultimately the destination of the message is already known. This means it is not necessary to route through the overlay exclusively until the server is reached; once an arbitrary peer that still has a functioning connection to the latter is found, delivery can proceed through the underlay from there on. This is desirable considering that every further hop once a working path is available only increases delay without any benefit. Based on this idea, for direct rerouting messages are simply forwarded through the overlay until a peer can deliver them. A flooding-based algorithm is likely to find such a peer faster as it can search along multiple paths at once at the price of higher bandwidth consumption and possibly redundant message delivery. Random walking will not cause additional traffic but is likely to take longer when forwarding over multiple hops. All this also applies to the indirect method, but with a modified procedure: First find a peer p that can reach server s, e.g., by using flooding or random walk. Then send messages via p to s. Since not the data messages themselves are used but only requests on who can reach s, the overhead is not as large as it would be for same technique with direct rerouting, thus making flooding more attractive for this approach.

Replication: In unstructured systems data replication is realized by distributing data to multiple peers. Due to the random nature of the overlay there is no way to deterministically find out where a certain data object will be replicated. The server has to search for the data. Unlike for rerouting, here it's not possible to take any shortcuts, because replication is realized on top of the overlay. Reactive replication will start forwarding its messages to its neighbors once a failure has been detected. Additionally, it will send a notification to the server via overlay, causing him to start searching. Since it takes some time until new data has spread across a system with many peers and reasonable bandwidth limits, the time until the failure is detected, plus the time until the server has received the notification, plus the time for the server to find it while still sparsely replicated is likely to exceed SCADA timeliness constraints. Proactive replication will require a constant amount of background traffic and memory. [28] shows optimal strategies on how to select peers for replication to minimize search time and maximize discovery rate. Nevertheless, replication in other than small-scale unstructured systems seems to be a poor choice when compared to rerouting as it takes longer, is more complicated and yet consumes more bandwidth and memory.

Applications for Structured P2P Systems. *Rerouting:* The general premises and features established for unstructured systems hold here. Since in structured networks peers are addressed by routing towards them, a proposed algorithm for direct rerouting is to forward a message along its overlay path to its destination and checking for an available underlay connection at every hop to directly deliver it, well aware that for exclusive overlay routing this would lead to congestion close to the destination. Indirect rerouting, with sender p, receiver s and message m could be implemented by first looking up peer q, who is responsible for m, and then sending m via q to s. While for pure overlay rerouting unstructured systems would clearly outperform structured ones, this is not the case here. Getting closer to the destination with every hop in the overlay does not relate to the probability of encountering a node with a non-broken underlay path to the former.

Replication: Replication in structured systems can be realized by using all peers in the overlay for distributed data storage, accessible over the DHT interface. As structured networks offer more functional possibilities, there are many possible algorithms. An example given in [21] for a reactive approach originated by the sender p of a data message m on failure detection works in three steps: First, m is stored in the DHT with its unique identifier. Then, the server s, which was the original recipient for m detects (through the SCADA application logic) the loss or delay of m. Finally, s can retrieve m from the DHT. Another schematic algorithm, this time for proactive replication: Every peer p deterministically picks i random surrogate peers $q_1, q_2, \ldots, q_i$, at which it replicates its data. As soon as a server can't reach p it instead tries to obtain the required data from the surrogates. In conclusion, structured systems are well suited for data replication since they have scalable upper bounds and offer built-in load balancing.

5 Mapping P2P Techniques to SCADA Systems

After discussing the general applicability of P2P technologies for SCADA protection, we now investigate how to select a specific P2P technique for a given SCADA system. Our P2P technology mapping onto SCADA systems is generic and considers the generic rerouting and data replication mechanisms and does not make any assumptions regarding scale, capabilities or topology of the SCADA system. Generally, it might not be appropriate to deploy a single P2P overlay on top of the whole SCADA system, e.g., to preserve locality of privacy of data. In such a case, different parts of the system have to be viewed and analyzed separately. Consequently, one should first identify the logical subsystems, and then select a suitable P2P technique for each subsystem.

5.1 Identification of Subsystems

There exist a number of mandatory criteria for partitioning the large-scale SCADA system: (a) Physical network topology as peers must be able to communicate etc, (b) security policies that may close off certain parts of a system to prevent external access or information leaks, and (c) ownership, as different parts of the system may be operated by different entities, thus should belong to different overlays. Additionally, included are legal obligations, regulations or political reasons. The resulting subsystems are unique, so a single one might be created for a number of reasons. Besides these non-negotiable aspects, further subsystems can be created based on the data flow model. It might be a bad idea to send operation-critical and non-critical message over the same overlay as both have very different requirements. The same goes for small-sized messages and larger ones. On the other hand, since many P2P properties like robustness increase with scale, overlays should be shared if possible. Fig. 4 shows an example of how multiple subsystems are identified within a generic SCADA hierarchy. The resulting subsystems are not necessarily disjoint, since points acting as bridges between two SCADA layers may be contained in two or more of those.

5.2 Selection of Appropriate P2P Techniques

Once a subsystem has been identified, a specific P2P technique can be selected, depending on the following aspects: (1) Scale, i.e., the number of peers meeting the P2P prerequisites, (2) requirements, i.e., certain properties the SCADA applications expect, e.g., timeliness constraints, communication model (push/pull), reliability etc, and (3) capabilities, i.e., peer resources and capabilities, i.e. memory, computational power, bandwidth, etc.

Assuming P2P is feasible, the first major choice is selecting the type of P2P application. Two possible options were discussed in Section 4.2, namely rerouting and data replication.

For rerouting, the use of a unstructured overlay is suggested, for the reasons given in Sections 3.2 and 4.2. In short these are:(1) High overlay flexibility w.r.t. to minimum requirements, node heterogeneity, selection of neighbors etc, (2) the overlay itself is very scalable, only searching in it is not. For rerouting this is not an

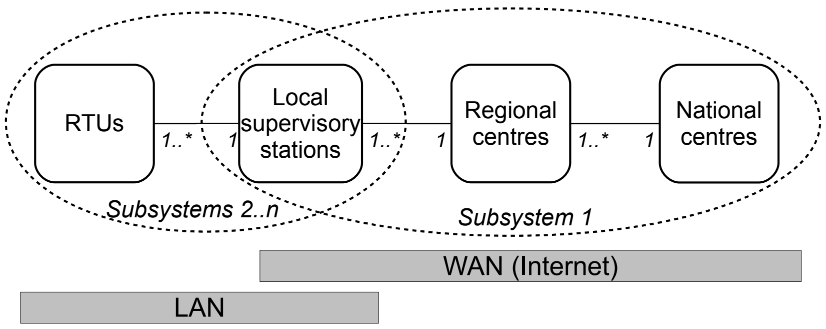

Fig. 4. Subsystem identification

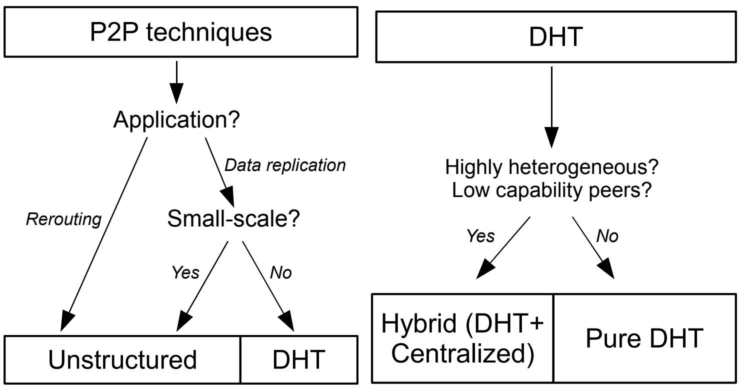

Fig. 5. Decision tree for basic technique

issue since the underlay is used as soon as possible, (3) a random topology which is hard to predict and robust against failures, and (4) Low maintenance overhead.

For data replication the case is more complex. Generally, DHT are preferred (see Section 4.2), however, for small-scale systems these make little sense since all properties regarding load balancing, neighbor selection etc. converge only for a high number of peers. Because of this, small-scale data replication is best realized with a highly connected unstructured overlay comparable to RON. Fig. 5 (left) summarizes the decision tree up to this point.

Because DHT don't support heterogeneous peers, the peers with the lowest capacities set the upper bound. So if there happens to be a big gap in terms of capabilities between the peers, a hybrid system should be considered, where the weaker peers are being managed by the stronger ones. This generally holds for situations where a minority thwarts overall performance.

In Section 3.2 three basic classes were identified, with respectively high, medium or low needs. To decide which concrete DHT to use one has to balance between requirements, capabilities and scale. Assuming that performance should be prioritized in terms of routing latency and robustness, this is achieved by utilizing as much capacities as possible. In Fig. 5 (right) the decision tree is completed for DHT.

6 Conclusions and Future Work

This paper provided a comprehensive analysis and classification of existing P2P technologies and their abilities to fulfil a SCADA system's needs. The pitfalls when using P2P techniques in a SCADA context were identified and a set of guidelines and best practices on how to avoid them were established. Previously designed P2P approaches were integrated into a taxonomy and their algorithmic principles were compared to those of other, newly proposed methods. To map specific P2P technologies to a given system, it was first shown how to partition it into different classes. The selection of appropriate techniques for each of those classes was based on previously drawn conclusions. This paper developed basic rules and concepts for the combination of SCADA and P2P. Furthermore, the analysis of P2P technologies for these specific types of applications provides a foundation for future research such as the application of the described methods for concrete SCADA systems to gain additional insights. Furthermore, trustworthiness measures should be developed to quantify and assess the overall increase of SCADA resilience through the P2P-enabling.

References

1. Beitollahi, H., Deconinck, G.: Analyzing the Chord Peer-to-Peer Network for Power Grid Applications. In: Fourth IEEE Young Researchers Symposium in Electrical Power Engineering (2008)
2. Beitollahi, H., Deconinck, G.: Peer-to-Peer Networks Applied to Power Grid. In: Proceedings of the International conference on Risks and Security of Internet and Systems, CRiSIS (2007)
3. Deconinck, G., Vanthournout, K., Beitollahi, H., Qui, Z., Duan, R., Nauwelaers, B., Lil, E., Driesen, J., Belmans, R.: A Robust Semantic Overlay Network for Microgrid Control Applications. In: Proceedings of the Workshop on Software Architectures for Dependable Systems, WADS (2008)
4. Fernandez, J.D., Fernandez, A.E.: SCADA Systems: Vulnerabilities and Remediation. Journal of Computing Sciences in Colleges 20(4) (2005)
5. Krutz, R.L.: Securing SCADA Systems (2005)
6. Lua, K., Crowcroft, J., Pias, M., Sharma, R., Lim, S.: A Survey and Comparison of Peer-to-Peer Overlay Network Schemes. IEEE Communications Surveys and Tutorials 7(2) (2005)
7. Androutsellis-Theotokis, S., Spinellis, D.: A Survey of Peer-to-Peer Content Distribution Technologies. ACM Computing Surveys, 36(4) (2004)
8. The Gnutella Protocol Specification v0.4 (2000), `http://www.stanford.edu/class/cs244b/gnutella_protocol_0.4.pdf`
9. Chawathe, Y., Ratnasamy, S., Breslau, L., Lanham, N., Shenker, S.: Making Gnutella-like P2P Systems Scalable. In: Proceedings of the 2003 ACM SIGCOMM Conference (2003)
10. Reverse Engineered FastTrack Protocol Specification, `http://cvs.berlios.de/cgi-bin/viewcvs.cgi/gift-fasttrack/giFT-FastTrack/PROTOCOL?revision=1.19`
11. Stoica, I., Morris, R., Karger, D., Kaashoek, F.M., Balakrishnan, H.: Chord: A Scalable Peer-to-Peer Lookup Service for Internet Applications. In: Proceedings of the 2001 ACM SIGCOMM Conference (2001)

12. Ratnasamy, S., Francis, P., Handley, M., Karp, R., Schenker, S.: A Scalable Content-Addressable Network. In: Proceedings of the 2001 ACM SIGCOMM Conference (2001)
13. Zhao, B.Y., Huang, L., Stribling, J., Rhea, S.C., Joseph, A.D., Kubiatowicz, J.D.: Tapestry: A Resilient Global-Scale Overlay for Service Deployment. IEEE Journal on Selected Areas in Communications, 22(1) (2004)
14. Maymounkov, P., Mazières, D.K.: A Peer-to-Peer Information System Based on the XOR Metric. In: Proceedings of the 2nd International Workshop on Peer-to-Peer Systems, IPTPS (2002)
15. Rowstron, A.I.T., Druschel, P.: Pastry: Scalable, Decentralized Object Location, and Routing for Large-Scale Peer-to-Peer Systems. In: Proceedings of the IFIP/ACM International Conference on Distributed Systems Platforms, Middleware (2001)
16. Kaashoek, F., Karger, D.R.: Koorde: A Simple Degree-Optimal Distributed Hash Table. In: Kaashoek, M.F., Stoica, I. (eds.) IPTPS 2003. LNCS, vol. 2735, Springer, Heidelberg (2003)
17. Gupta, I., Birman, K., Linga, P., Demers, A., van Renesse, R.: Building an Efficient and Stable P2P DHT through Increased Memory and Background Overhead. In: Kaashoek, M.F., Stoica, I. (eds.) IPTPS 2003. LNCS, vol. 2735, Springer, Heidelberg (2003)
18. Zhuang, S.Q., Zhao, B.Y., Joseph, A.D., Katz, R.H., Kubiatowicz, J.D.: Bayeux: An Architecture for Scalable and Fault-Tolerant Wide-Area Data Dissemination. In: Proceedings of The International Workshop on Network and Operating Systems Support for Digital Audio and Video, NOSSDAV (2001)
19. Kubiatowicz, J., Bindel, D., Chen, Y., Czerwinski, S., Eaton, P., Geels, D., Gummadi, R., Rhea, S., Weatherspoon, H., Wells, C., Zhao, B.: OceanStore: An Architecture for Global-Scale Persistent Storage. In: Proceedings of the international conference on Architectural support for programming languages and operating systems (ASPLOS), vol. 28 (2000)
20. Castro, M., Druschel, P., Kermarrec, A., Rowstron, A.: SCRIBE: A Large-Scale and Decentralized Application-Level Multicast Infrastructure. IEEE Journal on Selected Areas in communications 20(8) (2002)
21. Germanus, D., Khelil, A., Suri, N.: Increasing the Resilience of Critical SCADA Systems Using Peer-to-Peer Overlays. In: Proc. of The 1st International Symposium on Architecting Critical Systems, ISARCS (2010)
22. Dongni, R., Li, Y.T.H. and Chan, S.H.G. On reducing mesh delay for peer-to-peer live streaming
23. Guillaume, J.L., Latapyand, M., Magnien, C.: Comparison of Failures and Attacks on Random and Scale-Free Networks. In: Anderson, J.H., Prencipe, G., Wattenhofer, R. (eds.) OPODIS 2005. LNCS, vol. 3974. Springer, Heidelberg (2006)
24. Dinger, J., Hartenstein, H.: Defending the sybil attack in p2p networks: taxonomy, challenges, and a proposal for self-registration. In: Proceedings of The First International Conference on Availability, Reliability and Security, ARES (2006)
25. Singh, A., Castro, M., Druschel, P., Rowstron, A.: Defending against eclipse attacks on overlay networks. In: Proceedings of the ACM SIGOPS European Workshop, EW (2004)
26. Saxena, N., Tsudik, G., Yi, J.H.: Admission control in peer-to-peer: design and performance evaluation. In: Proceedings of the 1st ACM workshop on Security of ad hoc and sensor networks, SASN (2003)
27. D'Antonio, S., Romano, L., Khelil, A., Suri, N.: INcreasing Security and Protection through Infrastructure REsilience: the INSPIRE Project. In: Proceedings of The Workshop on Critical Information Infrastructures Security, CRITIS (2008)
28. Cohen, E., Shenker, S.: Replication Strategies in Unstructured Peer-to-Peer Networks. In: Proceedings of the 2002 ACM SIGCOMM Conference (2002)