Denial-of-Service in Automation Systems

Wolfgang Granzer, Christian Reinisch, Wolfgang Kastner *
Vienna University of Technology, Automation Systems Group
{w,cr,k}@auto.tuwien.ac.at

Abstract

Security aspects of today's automation systems gain increasing importance. One critical point regarding security is the exchange of control data over the network. Recently, cryptographic techniques have been developed that can protect the transmitted data against a malicious interference. However, there are security threats which cannot be prevented using cryptographic techniques. Typical representatives are Denial-of-Service (DoS) attacks. Therefore, this paper presents a novel, generic approach how such DoS attacks can be prevented or, if prevention is not possible, can be detected at least.

1. Introduction

Functions of today's automation systems are normally realized by distributed control applications. As it is naturally the case in these systems, there is an inherent need to exchange control data over the network. An important step towards a secure automation system is to protect the transmitted control data against malicious interference.

According to [7], the exchanged control data has to be protected against the security threats *interception, fabrication, modification*, and *interruption*. A protection against interception (e.g., network sniffing), modification (e.g., man-in-the-middle attacks), and fabrication (e.g., replay attacks) can be achieved by guaranteeing *integrity, freshness*, and *confidentiality* [5] of the transmitted control data. To achieve these security objectives, a secured channel has to be provided. This secured channel can be established by using physical and/or cryptographic techniques. Possible solutions to secure the transmitted control data in automation systems can be found in [8, 5].

Nevertheless, there are security threats which cannot be prevented using cryptographic methods. Typical representatives of such threats are *Denial-of-Service (DoS) attacks* which have the objective of making a service or data unavailable (*interruption*). To interrupt the communication in a network, the adversary tries to waste network resources (e.g., by flooding the network with unsolicited messages) and/or system resources (e.g., by consuming processor time of a server by sending multiple requests) to prevent the target from performing its expected function. DoS attacks foremost have massive economic impact. Consider, for example, an assembly line that is the target of a DoS attack. A shutdown of the line will lead to an economic impact that can be easily compared to the impact of a successful attack on the company Web server – the only difference being that, for the Web server, elaborate IT security measures are already common practice.

Therefore, DoS attacks are to some extent considered as a serious threat also for automation systems [2, 1]. However, DoS attacks are hard to handle especially in embedded networks. The main reason is that adversaries may dispose of much more processing power than embedded devices typically found in automation systems (*resource gap*). Therefore, it is easy to exhaust the limited resources available at the embedded devices [10]. Consider, for example, an adversary with a standard PC gaining access to a network that consists of small microcontrollers. The situation is further aggravated if multiple adversaries conjointly attack the same system resource (*distributed DoS*).

To counteract DoS attacks, different approaches exist [4]. However, since they are mostly designed for the use in the IT world, they cannot be directly mapped to automation systems without proper adjustments. The reasons are the limited system resources of the embedded field devices. Moreover, most countermeasures were not designed for the use in non IP networks. Therefore, the rest of this paper presents schemes and mechanisms that can be used to handle DoS attacks in automation systems.

2. DoS in automation systems

DoS attacks can be classified into two different categories. On the one hand, an adversary may try to interrupt the operation of a single network node by wasting its system resources (*host-based DoS attacks*). Typical targets of host-based DoS attacks are devices critical for system operation (e.g., key servers, firewalls), devices that concentrate automation functionality (e.g., controllers, operator workstations) or interconnection devices such as gateways and routers that provide an interconnection to other, possibly foreign networks. On the other hand, the network itself may be target of DoS attacks (*network-based DoS*

[*] This work was funded by FWF (Österreichischer Fonds zur Förderung der Wissenschaftlichen Forschung; Austrian Science Foundation) under the project P19673.

1-4244-1506-3/08/$25.00 ©2008 IEEE

attacks). In this case, the adversary tries to waste the network bandwidth to completely interrupt the communication between the devices located in the affected segment.

In both cases, two possibilities exist to handle DoS attacks: *DoS prevention* and *DoS detection*. DoS prevention has the aim to limit the access to system resources in a way that an adversary does not have the opportunity to successfully perform DoS attacks. One opportunity to fully prevent DoS attacks is to limit the physical access to the network medium and to the devices that have an interface to the medium. In a wired network, this can be done by immuring the network cable or by locking the devices into a safe containment (*physical security*). Obviously, such an isolation is not always easy to achieve. For example, in case of wireless or powerline networks, an isolation is impossible due to the openness of the medium. Furthermore, a DoS prevention is hard to achieve if the adversaries have much more processing power than the victims.

Therefore, in situations where prevention methods are inapplicable, DoS attacks shall at least be detected. Detecting methods try to discover an abnormal system behavior. After detection, it must be determined whether the abnormal situation may lead to a DoS attack. If a DoS attack is suspected, countermeasures have to be initiated to minimize the consequences, i.e., to avoid propagation to other, not yet infected, network segments.

To combine the advantages of both methods, a hybrid approach is the most appropriate solution for automation systems. Therefore, such a combination of prevention and detection mechanisms is presented in the next section.

3. A common approach to handle DoS in automation systems

The presented solution uses a combination of DoS prevention and DoS detection. This scheme works as follows: On the one hand, system-critical devices are protected against host-based DoS attacks whenever possible. To achieve this, a prevention mechanism based on client puzzles is implemented on these devices (cf. Section 3.1). On the other hand, network-based as well as host-based DoS attacks that cannot be prevented with reasonable effort (e.g., that are targeted at low performance embedded devices) shall at least be detected and adequate countermeasures shall be initiated (cf. Section 3.2).

Fig. 1 shows an example of an automation network where the above mentioned approach is used.[1] The intention is to protect web gateways, operator workstations, and controllers against host-based DoS attacks using a prevention mechanism. Host-based attacks where a prevention is not possible as well as network-based ones are detected by the routers and gateways.

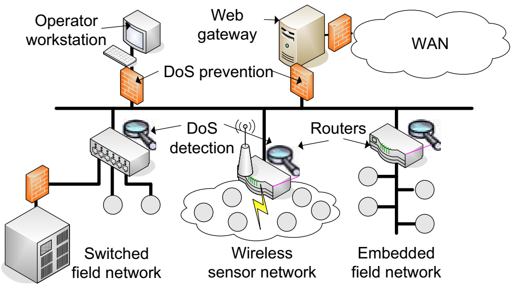

Figure 1. DoS in automation systems

3.1. DoS prevention using client puzzles

To prevent host-based DoS attacks, a client (e.g., operator workstation) that wants to set up a secured channel to a server (e.g., controller) has to prove its identity by using a two phase authentication protocol. In the first phase (*pre-authentication phase*), a common technique called client puzzle is used [3]. The main objective of a client puzzle is to make a DoS attack at least as expensive for the adversary as for the target in terms of computational cost. The following client puzzle has to be solved by the client [9]:

$$h(C, N_s, N_c, X) = \underbrace{000...0}_{k\ first\ bits}\ Y$$

whereas h denotes a secure hash function, C the identity of the client, N_s the server's nonce, N_c the client's nonce, k the number of the hash bits that have to be 0 (i.e., the difficulty of the puzzle) and X the solution of the puzzle. The client has the objective to find the input value X which produces this hash value by brute force. As it is very easy to verify whether the solution is valid or not (e.g., by comparing the solution with pre-calculated puzzle solutions stored in a lookup table), the client must pay more computing costs than the server. If the client has solved the puzzle, the server returns an authentication ticket. Using this ticket, the client has exactly one attempt to pass the second authentication phase (*secure authentication phase*). In this phase, the client proves its identity using, for example, a secured password, a client certificate, or a shared secret key. If the authentication attempt fails, it has to solve another client puzzle. This scheme has the advantage that if the client has not solved the client puzzle, the server does not need to consume neither memory space nor processing power since puzzles and their solutions can be stored in simple lookup tables. Another advantage is that brute force attacks on passwords or secret keys are prevented since the attacker has only one authentication attempt after a puzzle has been solved. Furthermore, the additional effort for legitimate devices is relatively small since only one puzzle has to be solved to set up a secured channel.

The required complexity of the client puzzle (i.e., in this case the value of k) depends on the computational power of the involved network nodes. However, in embedded field networks, the used microcontrollers have insufficient hardware resources to solve this puzzle within

[1] Obviously, the illustrated automation network is only a theoretical example since such a combination of switched, wireless and line topology is uncommon.

an adequate time. Therefore, DoS prevention using client puzzles is implemented on system-critical devices where the connecting clients are supposed to have sufficient processing power. Typical examples are the web interface of a web gateway as well as the management interface of key servers, gateways, and controllers.

3.2. DoS detection

One possibility to detect DoS attacks is the use of a so called *intrusion detection system (IDS)*. The objective of an IDS is to detect abnormal system behavior, e.g., abnormal network traffic or abnormal device behavior. According to [6], an IDS commonly consists of four components:

- *Data gathering component* responsible for collecting the data by observing the network traffic.
- *Data processing component* concerned with processing the collected data and deciding whether abnormal behavior is present.
- *Data storage unit* in charge of collecting the results and storing the observed data (communication traces).
- *Response unit* responsible for initiating actions to minimize the consequences of an attack.

In the IT domain, many different IDS solutions exist. However, their data gathering units are mainly designed for IP networks and the assumed system model is significantly different from the process model of an automation system. Therefore, IDS from the IT world can only be used as a starting basis – parts have to be redesigned. The functionality of the IDS itself is incorporated into the software implementation of routers and gateways.

3.3. DoS countermeasures

After a DoS attack has been detected by the IDS, it is essential to minimize the resulting consequences. The most effective solution is to stop the adversary from attacking the target. To achieve this, the source(s) of the DoS attack have to be identified and isolated from the rest of the network. This keeps the system operable and prevents a propagation of the DoS attack.

The most appropriate way to isolate the source(s) of the DoS attack is closely related to the physical topology of the affected network segment. As shown in Fig. 1, star (e.g., switched Ethernet networks), line, and even free topology are common in automation networks.

In wired star topologies, an isolation can be accomplished by cutting the communication line to the source(s) of the attack. For example, in a switched Ethernet network, the port where a DoS attack has been detected can simply be deactivated.

In wireless networks, the isolation highly depends not only on the logical topology (e.g., star, mesh) but also on the used communication model (e.g., peer-to-peer or coordinator-to-peer). The basic idea is to isolate an affected zone and to find routes that bypass this zone [10]. However, this is currently work-in-progress and therefore not elaborated further in this paper.

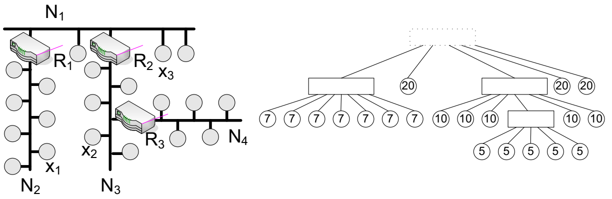

Figure 2. Device isolation

Finally, in wired line or free topology, an isolation can only be achieved by decoupling the whole network segment. This action has to be executed by the interconnection device located at a network segment border. Consider, for example, device x_1 in Fig. 2 starts flooding attack. If this misbehavior is detected by the IDS module running in router R_1, the router can isolate network segment N_2 by stopping forwarding messages to N_1. Therefore, the rest of the network (i.e., N_1, N_3, and N_4) can continue normal operation. The main drawback of this solution is that all network members of N_2 become isolated. To estimate the damage a device may cause when performing a DoS attack, we define the so called *DoS risk factor (DoS-RF)*. To calculate the DoS-RF, the network topology is represented as tree based graph. On the right hand side of Fig. 2, the corresponding graph of the example network is shown. While interconnection devices are represented as rectangular nodes, field devices are presented as circular nodes. An edge between two nodes stands for a physical network connection between two devices. Furthermore, the sequence of the children of an interconnection node defines the physical location of the device within the network segment – the left-most child is always the device located next to the interconnection device[2]. The DoS-RF of a device x is now calculated as the sum of all node weights of all children and sub-children of the parent of the device x. The weight represents the damage that occurs if a device fails due to a security attack. In a first approach, the amount of data points a device holds as well as their types (e.g., safety/security critical, system, normal, low) are used to determine these node weights. Consider, for example, device x_2: to isolate this device, the routers R_2 and R_3 have to decouple N_3. This means that all members of N_3 are no longer able to communicate with N_1, N_2, and N_4. However, since there is no route from N_4 to N_1 and N_2, also the members of N_4 are only able to communicate with each other. Any communication with the other remaining, operable network segments is interrupted. Because of the severity of an attack on node x_2, its DoS-RF equals 10, which is the weight count of all children and sub-children of its parent node R_2.[3]

While a low DoS-RF indicates that decoupling the device from the rest of the system influences only a smaller part of the network, a high DoS-RF represents the oppo-

[2] It is assumed that the network topology is following a line topology. However, if there is free topology, the topology has to be converted to a line topology by inserting virtual interconnection nodes.

[3] It is assumed that all devices hold a single data point of type ”low” and therefore each have a weight of 1.

site. In case of an attack, a large number of devices would become isolated. To counter large DoS-RFs, a physical network segment has to be divided into smaller so called *virtual network segments*. Virtual network segments are not logically separated from each other i.e., they do not have a dedicated network address. Therefore, virtual network segments are invisible to network members. Dividing a physical network segment into two virtual network segments is done by placing a so called *virtual bridge* at these virtual segments' intersection point. A virtual bridge has two (or more) network interfaces and simply forwards incoming messages to the other network interface. However, in contrast to a layer 2 bridge, it is possible to request the virtual bridge to decouple virtual network segments by cutting the communication line between them. Consider, for example, device x_3: in the original network (cf. Fig. 2), x_3 has a DoS-RF of 20. However, by placing a virtual bridge between R_2 and x_3, the DoS-RF of x_3 can be reduced to 2. This is due to the fact that now also the virtual bridge is able to decouple x_3 and the device next to x_3 (cf. Fig. 3).

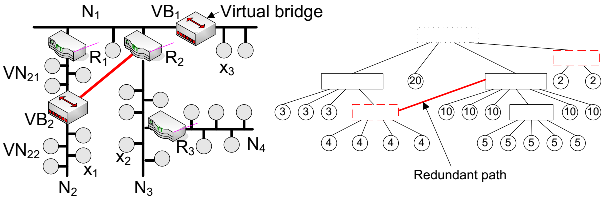

Figure 3. Virtual bridges

This solution has one drawback: if a (virtual) network segment is located between two other (virtual) network segments, decoupling it also isolates all succeeding network segments from the remaining network. Consider, for example, the virtual bridge VB_2 that is added in N_2. This virtual bridge reduces the DoS-RF of all succeeding devices to 4. However, the devices located between R_1 and VB_2 still have a DoS-RF of 7 since a DoS attack within VN_{21} also decouples VN_{22} from the remaining network. To avoid this, alternative communication paths have to be provided by using redundant interconnections between virtual bridges and interconnection devices. Suppose, for example, a redundant connection is added between VB_2 and R_2. Using this connection, the members of VN_{22} are still able to communicate with the remaining network. In this case, the DoS-RF of the members of VN_{21} is reduced from 7 to 3. However, adding redundant connections may introduce cyclic paths. Therefore, a mechanism is necessary that only enables these redundant connections when they are required. A possible realization includes a dedicated protocol that caters for communication among all interconnection devices and virtual bridges. Thus, information on attacks can be exchanged and the most appropriate countermeasures can be triggered. An alternative solution would be to permanently enable redundant connections and to discard all duplicate messages. Furthermore, adding redundant paths may introduce additional risk. However, this risk can be eliminated by limiting the physical access to the connection (e.g., immuring the redundant connection), since redundant paths are normally point-to-point connections.

The placement of virtual bridges and redundant paths is not trivial. Therefore, it has to be foreseen at installation time by the project engineer. To assist the project engineer, engineering tools shall help to find critical points (i.e., devices with high DoS-RFs) in the network topology and suggest where to add virtual bridges and redundant connections best.

4. Conclusion and future work

Providing an effective protection against DoS attacks is by no means a trivial task, especially in embedded field networks. Therefore, a generic approach how DoS attacks can be handled in automation systems has been described in this paper. As a next step, the presented approach will be refined. This includes a detailed design of the IDS and the communication protocol that manages an isolation and redundant path activation. Furthermore, the DoS prevention and detection mechanisms shall be evaluated with the help of a prototype implementation and a simulation framework of a secure building automation system [5]. Finally, the automatic generation of tree graphs out of the engineering tools will be assessed.

References

[1] Guide to Industrial Control Systems (ICS) Security. NIST SP800-82, 2007. Second Public Review Draft.

[2] Power System Control and Associated Communications - Data and Communication Security. IEC 62351, 2007.

[3] C. L. Bowen, T. K. Buennemeyer, and R. W. Thomas. Next Generation SCADA Security: Best Practices and Client Puzzles. In *Proc. 6^{th} IEEE Workshop on Systems, Man and Cybernetics*, pages 426–427, 2005.

[4] K. J. Cox and C. Gerg. *Managing Security with Snort and IDS Tools*. O'Reilly & Associates, 2004.

[5] W. Granzer, C. Reinisch, and W. Kastner. Key Set Management in Networked Building Automation Systems using Multiple Key Servers. In *Proc. 7^{th} IEEE International Workshop on Factory Communication Systems*, pages 205–214, 2008.

[6] C. Krügel. *Network Alertness: Towards an Adaptive, Collaborating Intrusion Detection System*. PhD thesis, Vienna University of Technology, 2002.

[7] S. L. Pfleeger and C. P. Pfleeger. *Security in Computing*. Prentice Hall, 3^{rd} edition, 2002.

[8] A. Treytl, T. Sauter, and C. Schwaiger. Security Measures for Industrial Fieldbus Systems - State of the Art and Solutions for IP-based Approaches. In *Proc. 5^{th} IEEE International Workshop on Factory Communication Systems*, pages 201–209, 2004.

[9] A. Tuomas, N. Pekka, and L. Jussipekka. DOS resistant authentication with client puzzles. In *Proc. International Workshop on Security Protocols*, pages 170–177, 2000.

[10] A. D. Wood and J. A. Stankovic. Denial of Service in Sensor Networks. *IEEE Computer*, 35(10):54–62, 2002.