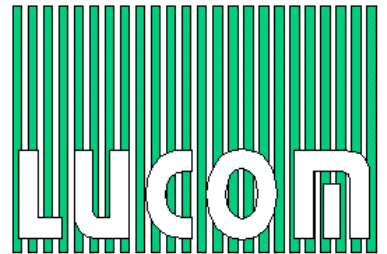


# Innominate **mGuard**

## Industrial Network Security



Innominate  
**certified**  
partner



Innominate  
**mGuard**

**[WWW.LUCOM.DE](http://WWW.LUCOM.DE)**

**LUCOM** GmbH

Komponenten & Systeme  
Ansbacher Str. 2a

**D 90513 Zirndorf**

Tel. +49 (0) 9127 / 59 460-10

Fax. +49 (0) 9127 / 59 460-20

E-Mail: [info@lucom.de](mailto:info@lucom.de)



## Application note

# Innominate mGuard rollout support

## Table of contents

1) Introduction.....	3
2) Roll-out support framework.....	4
2.1) Script rollout.sh.....	4
2.1.1) Bootstrap in detail.....	4
2.1.2) Remarks.....	4
2.2) Using ISCM for supporting the roll-out process.....	5
2.2.1) Settings.....	5
2.2.2) Generate the configuration files.....	5
3) Appendix A – Reference script 1 (tftp, dynamic).....	6
4) Appendix B – Reference script 2 (tftp, static).....	7
5) Appendix C – Reference script 3 (wget, dynamic).....	8
6) Appendix D - Further readings.....	9
6.1) gaiconfig.....	9

# 1 Introduction

The following actions have to be taken, when putting an Innominate mGuard into operation:

- changing the root password
- changing the administrator password
- open SSH port to allow the configuration manager to access the Innominate mGuard
- allow remote administration (via Web Interface)
- (generate certificate)
- configuration of IP-addresses
- configure Stealth/Router Mode
- Generate keypairs and distribute the public key for passwordless SSH-authentication
- configure firewall rules

When rolling out projects with lots of devices it could be very time consuming to configure each mGuard one-by-one using the Web-interface. This document describes ways to partially automate the roll-out process.

## 2 Roll-out support framework

### 2.1 Script *rollout.sh*

The Innominate mGuard provides basic means for supporting the rollout process<sup>1</sup>. The implementation of the process itself depends on the user and is therefore not part of the standard-distribution.

After flashing the firmware (see Innominate mGuard manual) the Innominate mGuard checks for a file *rollout.sh* in the same directory on the PC where also the Innominate mGuard image files are located. In case the file is present, it is downloaded to the Innominate mGuard and executed. Since the script is unsigned, the Innominate mGuard needs to be flashed before to erase all confidential information like keys or passwords.

The file *rollout.sh* should contain a Bourne shell-script. The script can be used to retrieve the configuration data for the Innominate mGuard from the PC, and to start the Innominate mGuard configuration binary (*gaiconfig*), which will configure the Innominate mGuard accordingly. Chapter 2.1.1 contains more detailed information on the bootstrap process and on how to implement the script.

The contents of the script depend on the particular requirements of the user. Therefore the script is not provided by Innominate. The roll-out support can be implemented in a way that all devices get the same configuration (see Appendix B), or it can be implemented dynamically, configuring each device depending on its serial or flash-number (see Appendix A and C).

Appendix A, B and C contain example scripts for different scenarios. Use these scripts as a reference for your implementation.

#### 2.1.1 Bootstrap in detail

- 1) Every bootstrapping filesystem-image (*jffs2.img.p7s*) has a directory called */bootstrap/* that contains the bootstrapping control-script (*/bootstrap/install.sh*). The control script deletes the entire */bootstrap* directory after the bootstrap process is successfully finished. Everytime the kernel boots, it checks for an unfinished bootstrapping process by checking the existence of */bootstrap*.
- 2) The *install.p7s*-script loads a script called *rollout.sh* from the tftp-server after loading and flashing the jffs2 filesystem image.  
*rollout.sh* is written to */bootstrap/rollout.sh* of the flash-filesystem and executed from there. A missing *rollout.sh* script on the tftp-server is ignored.
- 3) The *rollout.sh* script has to create the */bootstrap/preconfig.sh*-script. This script will be executed at the end of the bootstrap process. An example of dynamic (database controlled) configuration depending on the flash-id or the serial number can be found in Appendix A or Appendix C.
- 4) The system starts bootstrapping after it has loaded and flashed *jffs2.img.p7s* completely, which is controlled by */bootstrap/install.sh*. After the installation has been successfully completed it checks the existence of the configuration-script */bootstrap/preconfig.sh*. If it does not exist, no further action is taken and the system indicates success by blinking with all 3 LEDs at once. If the script exists it is executed and the final blinkcode is changed, such that the status LED is inverted to show the successful execution of */bootstrap/preconfig.sh*.<sup>2</sup>

#### 2.1.2 Remarks

- *gaiconfig* can be called multiple times in *rollout.sh*, first for a master-configuration (e.g. *default.atv*) which contains a basic configuration to be used by all devices and second for the specific configuration *preconfig.atv*.
- *rollout.sh* could also prepare the Innominate mGuard for passwordless authentication by

<sup>1</sup> Innominate mGuard Release 2.1.3 or higher.

<sup>2</sup> During the execution of *rollout.sh* the network is available, during the execution of *preconfig.sh* the Innominate mGuard is in stand-alone mode.

- changing the trusted host of the Innominate mGuards or
- copy the public key of the user to the Innominate mGuard (user based authentication)

## 2.2 Using ISCM for supporting the roll-out process

ISCM can be used to generate the configuration files for the Innominate mGuard. ISCM does not support all variables of the Innominate mGuard. Please refer to the ISCM user manual to get an overview over the supported variables.

### 2.2.1 Settings

Before generating the configuration files with ISCM, please follow the steps below:

- Select for each Innominate mGuard **Local Host** as upload method in the menu **Upload configuration** (Device property window, please refer to the ISCM manual)
- Enter the directory where the configuration files should be placed in the menu **General options -> Roll-out options** (Device property window, please refer to the ISCM manual).
- Enter an identifier for each device in the menu **General options -> Roll-out options** (Device property window, please refer to the ISCM manual), e.g. the serial number of the device. This identifier will be used as filename for the configuration file.

### 2.2.2 Generate the configuration files

The process to generate the configuration files with ISCM is as follows:

- 1) Draw the network topology with the configuration manager and configure each Innominate mGuard according to chapter 2.2.1.
- 2) Open the context menu in ISCM by right-clicking on an Innominate mGuard and select **Generic PEP actions -> Create roll-out DB**.
- 3) Select **All** in the window that opens and start the generation process.

ISCM creates a file *<identifier>.atv* for each Innominate mGuard in the specified directory. The generated files can be used to configure the Innominate mGuards as described in Appendix A.

### 3 Appendix A – Reference script 1 (tftp, dynamic)

This is a sample *rollout.sh* script using tftp for download of the configuration file from the PC and using different configurations for each device. The script assumes, that the configuration filename on the PC is *<serialnumber>.atv*. It downloads the configuration file using tftp, creates the script *preconfig.sh* that calls the Innominate mGuard-configuration binary (*gaiconfig*). *preconfig.sh* is automatically executed after all packets are installed successfully (see chapter 2.1.1).

```
#!/bin/sh -ex
# The IP address of the DHCP/TFTP server
# is supplied by install.p7s
server=$1
export PATH=/bin:/bootstrap
# /proc filesystem is needed
mount -t proc none /proc
SERIAL=`cat /proc/sys/mguard/parameter/oem_serial`
# /proc must be unmounted to enable unmounting of the jffs2 filesystem later.
umount /proc
# This is the filename of the user supplied configuration file
# on the host in the TFTP-server directory
cfg_name=${SERIAL}.atv
# fetch the configuration-file "preconfig.atv"
tftp -g -l /bootstrap/preconfig.atv -r $cfg_name ${server}
# create a small configuration-script that installs the
# configuration fetched from ${server}
cat >/bootstrap/preconfig.sh <<EOF
#!/bin/sh
modprobe param_dev
gaiconfig --silent --set-all < /bootstrap/preconfig.atv
EOF
# Make it executable. It will be executed after all packets
# are installed completely.
chmod 755 /bootstrap/preconfig.sh
```

## 4 Appendix B – Reference script 2 (tftp, static)

This is a sample *rollout.sh* script using tftp for download of the configuration file from the PC and using a standard configuration file for each device. The script assumes, that the configuration filename on the PC is *preconfig.atv*. It downloads the configuration file using tftp, creates the script *preconfig.sh* that calls the Innominate mGuard-configuration binary (*gaiconfig*). *preconfig.sh* is automatically executed after all packets are installed successfully (see chapter 2.1.1).

```
#!/bin/sh -ex
# The IP address of the DHCP/TFTP server
# is supplied by install.p7s
server=$1
# This is the filename of the user supplied static configuration file
# on the host in the TFTP-server directory
cfg_name=preconfig.atv
export PATH=/bin:/bootstrap
# fetch the static configuration-file "preconfig.atv"
tftp -g -l /bootstrap/preconfig.atv -r $cfg_name ${server}
# create a small configuration-script that installs the
# configuration fetched from ${server}
cat >/bootstrap/preconfig.sh <<EOF
#!/bin/sh
modprobe param_dev
gaiconfig --silent --set-all < /bootstrap/preconfig.atv
EOF
# Make it executable. It will be executed after all packets
# are installed completely.
chmod 755 /bootstrap/preconfig.sh
```

## 5 Appendix C – Reference script 3 (wget, dynamic)

This is a sample *rollout.sh* script using *wget* for download of the configuration from the PC and using different configurations for each device. The script assumes, that the script on the server-side (*preconfig.cgi* in this example) identifies the device using the serial number. Additionally to the serial number several informations about the device are transferred from the Innominate mGuard to the server. The script on the server side (*preconfig.cgi* in this example) is responsible for creating the script *preconfig.sh* which is sent to the Innominate mGuard and installed in the Innominate mGuard */bootstrap* directory. *preconfig.sh* calls the Innominate mGuard configuration binary (*gaiconfig*) and is automatically executed after all packets are installed successfully (see chapter 2.1.1).

```
#!/bin/sh -ex

# The IP address of the DHCP/TFTP server
# is supplied by install.p7s
server=$1
export PATH=/bin:/bootstrap

# /proc filesystem is needed
mount -t proc none /proc

# fetch information about the system and replace spaces by %20
FID=`cat /proc/sys/mguard/hw/factoryregister`
HW=`cat /proc/sys/mguard/hw/revision`
SW=`cat /etc/version`
SERIAL=`cat /proc/sys/mguard/parameter/oem_serial |sed 's/ /\%20/g'`
CPU=`cat /proc/sys/mguard/hw/processortype |sed 's/ /\%20/g'`
NAME=`cat /proc/sys/mguard/parameter/name |sed 's/ /\%20/g'`

# call the configuration CGI with all the information about us and
# store the result as /bootstrap/preconfig.sh
# The configuration in this example is done by a http-server on
# the DHCP/TFTP server host.
wget -O /bootstrap/preconfig.sh "http://$server/cgi-
bin/preconfig.cgi?FID=$FID&HW=$HW&SW=$SW&CPU=$CPU&NAME=$NAME&SERIAL=$SERIAL"

# Make it executable. It will be executed after all packets
# are installed completely.
chmod 755 /bootstrap/preconfig.sh

# /proc must be unmounted to enable unmounting of the jffs2 filesystem later.
umount /proc
```

The CGI-script has to retrieve the configuration from a database (or from configuration files) and should create a shell script *preconfig.sh* that calls the Innominate mGuard configuration binary (*gaiconfig*):

```
#!/bin/sh
modprobe param_dev
gaiconfig --silent --set-all <<EOF
NETWORK_HOSTNAME=mGuard0232
MY_LOCAL_IP=192.168.30.40

...
EOF
```

In the example the configuration values *192.168.30.40* and *mGuard0232* have to be retrieved from a database (or configuration files) on the PC depending on the device .

## 6 Appendix D - Further readings

For detailed information on the Innominate mGuard please refer to the mGuard User Manual.

### 6.1 gaiconfig

This chapter provides some more details about the *gaiconfig*-binary that is used in the example scripts.

The Generic Administration Interface's (GAI) purpose is to provide a user and system interface to configure Innominate's server appliances and other UNIX alike operating systems. *gaiconfig* is the command line tool to get and set variables in all configuration files managed by GAI. Services are stopped and started as defined in the registry before the program exits. Upon successful completion a value of 0 is returned. This command can be used by all users who are members of the group 'admin'. Here is a selection of *gaiconfig* commands:

- help      Print a short description of the command.
- get-all    Dumps all read-write variables on stdout, using the Attribute-Type-Value (ATV) Object Serialization language.
- set-all    Sets all write-only and read-write variables mentioned in the file in Attribute-Type-Value (ATV) Object Serialization language from stdin.
- silent     Just rewrite the configuration files but don't stop and start services.
- reboot    Reboot the device using the "reboot" command.

For further information on *gaiconfig* please refer to the mGuard GAI Manual.