Configuring SNMP

Simple Network Management Protocol (SNMP) is an Internet Standard protocol for collecting and organizing information about managed devices on IP networks and for modifying that information to change device behavior. SNMP is widely used in network management for network monitoring. SNMP exposes management data in the form of variables on the managed systems organized in a management information base (MIB) which describe the system status and configuration.[1] Starting with our 4.2.0 release, the 128T introduces support for SNMP traps as well as a limited set of information retrievable via SNMP GET.

The 128T implementation of SNMP relies on the *snmpd* process running within the host operating system. All configuration for SNMP, however, is done within the 128T's data model via its administrative interfaces (e.g., PCLI, GUI, NETCONF, etc.).

MIB Definitions

Starting with our software release 4.2.0 we provde MIB files that describe the 128T SNMP objects and traps available on the 128T device and how they are encoded. For ease of use these are installed on the device itself in this location:

```
/usr/share/snmp/128technology/
```

A Network Management System (NMS) can import these MIBs when managing the 128T appliance. On a Centos based NMS the latest mibs can be optionally installed by downloading the latest "128T-mibs" package from the 128T yum repoisitory.

The specific objects available in the MIB are described in more details in the the section "MIB Layout".

Basic Configuration

Configuring SNMP on the 128T is done on a per-router basis, and is done within the `router > system > services > snmp-server` branch of the configuration hierarchy. There are three areas of configuration required: the *protocol* configuration, the *access* configuration, and the *notification receiver* configuration.

Protocol Configuration

The protocol configuration requires three settings: setting `enabled` to `true` enables the SNMP agent on your 128T. You must also set the `protocol` to `v2c` (SNMPv2c is the only version of SNMP supported by 128T). Finally, configure the `port` on which the SNMP agent will listen for inbound requests; this is typically `161`, the well-known port for SNMP agent requests.

Access Configuration

The access configuration lets you create a set of "whitelist" addresses allowed to interact with the 128T SNMP agent (i.e., to send GET requests to the 128T). This represents your SNMP management platform.

The access configuration, configured within `access-control` in the `snmp-server` element, has three components:

- name: a unique name given to the access configuration element (this is the "key" for the configuration, to uniquely identify a whitelisted SNMP source)
- community: the community string to expect in requests from the requesting SNMP agent, as an authentication mechanism
- source: the IP address of the whitelisted device

```
access-control        my-nms-platform
   name       my-nms-platform
   community  public
   source     10.128.201.2
exit
```

> Note: if you intend on polling the 128T device via one of the *managed interfaces* (i.e., one that is configured within 128T as a forwarding interface), this will require additional configuration steps. This in turn may affect the source address that the SNMP daemon sees as requests arrive. See the section below on Polling SNMP.

Notification Receiver Configuration

The notification receiver configuration defines where the 128T will send SNMP information in the event of a system issue. This is configured within `notification-receiver` in the `snmp-server` element. Like the access configuration, it too has three components:

- ip-address: the address of your trap receiver
- port: the UDP listening port on the trap receiver (typically 162)
- type: either `trap` or `inform`. This determines whether the 128T will send traps or informRequests to the receiver

Because the traps/informRequests are sent by the snmpd process running on the host Linux operating system, it is crucial that the host's routing table is capable of reaching the `notification-receiver` via the appropriate egress interface. Use the Linux command `ip r` to review the host's routing table to confirm that it meets your requirements.

Polling SNMP

Once the `snmp-server` is configured, the 128T device is eligible to be polled for SNMP requests from any source identified in an `access-control` configuration. Depending on your system architecture, this may require some additional configuration for polling to be successful, however.

Routing SNMP Responses

Because the SNMP protocol is handled by a system process running on the host operating system, it is important that your host operating system's routing table has routes suitable for returning responses to inbound queries to the appropriate destination(s) via the appropriate interface. Use the Linux command `ip r` to review the host operating system's route table, to ensure that snmpd is capable of transmitting responses to inbound polling requests.

> Note: this also applies to traps sent by the 128T to the `notification-receiver`.

When sendiing responses via a forwarding interface (as described below), the 128T software manages the Linux routing table rules on your behalf.

Polling via Forwarding Interfaces

A *forwarding interface* is one that is identified in your 128T configuration as being managed by the 128T software – generally, this is a `device-interface` of type `ethernet`, with a `pci-address` on your platform's PCI bus. In order to successfully poll a 128T via a forwarding interface, there are several other configuration components required.

1. A `host-service` on the network-interface, with type set to `custom` and `transport` set to UDP/161 (or whichever port you've specified in your `snmp-server` configuration).
2. The `host-service` should include one or more `access-policy` statements to allow access by the SNMP polling device(s).
3. The `access-control` in the `snmp-server` should have its `source` set to `169.254.127.126` instead of the actual address of your SNMP polling server. This is due to the fact that the inbound SNMP requests will arrive at the Linux host operating system via a *kernel network interface* (KNI), which performs source NAT of the outbound packets sent to snmpd.

Sample Configuration (Basic)

```
snmp-server
    enabled                 true
    version                 v2c
    port                    161

    access-control          my-nms-agent
        name        my-nms-agent
        community   public
```

```
        source     10.128.201.2
    exit

    notification-receiver  10.128.201.2 162 trap
        ip-address  10.128.201.2
        port        162
        type        trap
    exit
exit
```

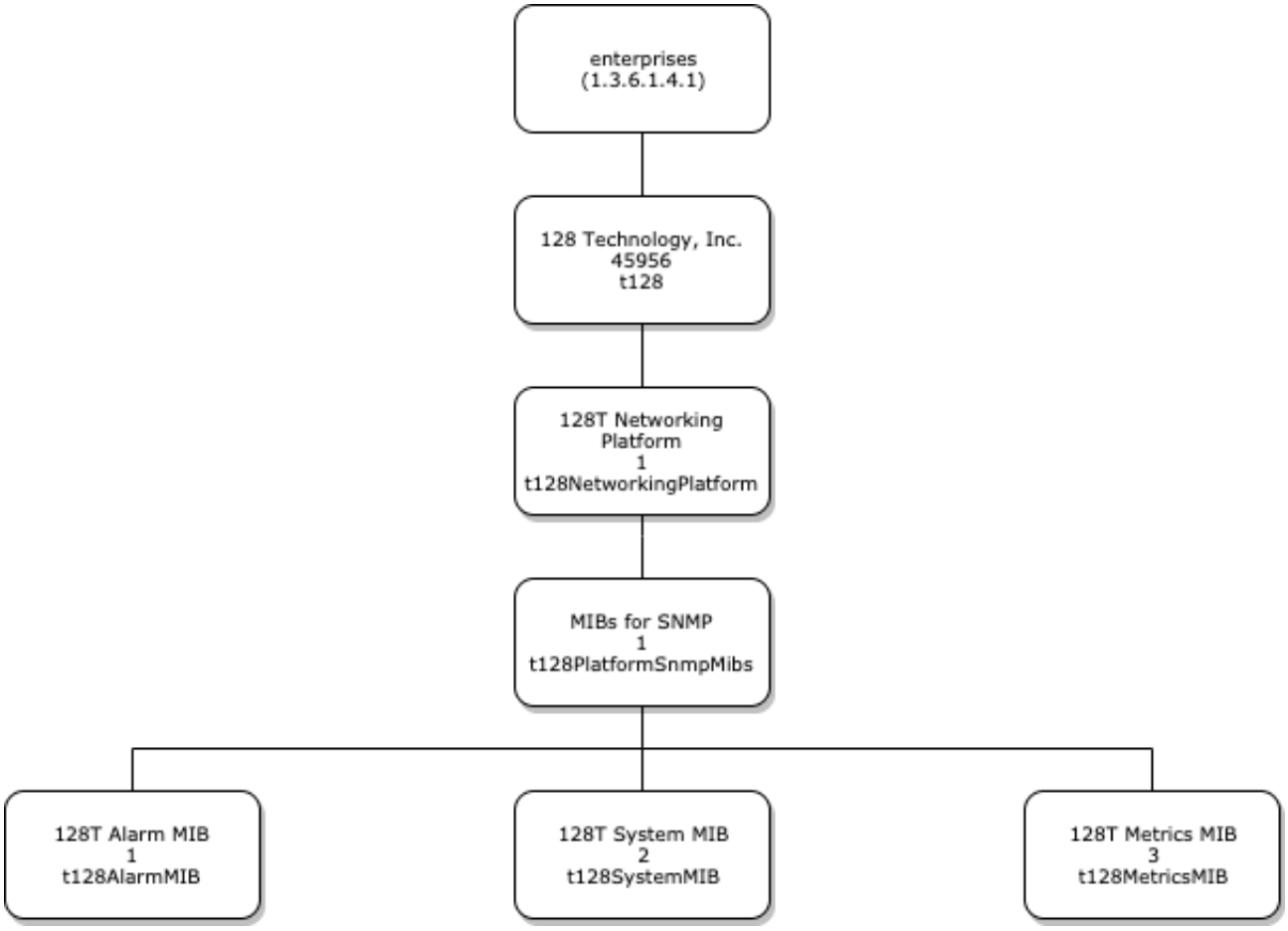
Sample Configuration (Via Forwarding Interface)

```
snmp-server
    enabled                true
    version                v2c
    port                   162

    access-control         snmp-agent
        name       snmp-agent
        community  public
        source     169.254.127.126
    exit
exit
network-interface
    name                   lan
    description            "LAN"

    inter-router-security  internal

    address                192.168.1.1
        ip-address     192.168.1.1
        prefix-length  24

        host-service   custom
            service-type   custom

            transport      udp
                protocol    udp

                port-range  161
                    start-port  161
                exit
            exit

            access-policy  10.128.201.2/32
```

```
                source      10.128.201.2/32
                permission  allow
            exit

            access-policy  blacklist
                source      blacklist
                permission  deny
            exit
        exit
    exit
exit
```

MIB Layout

The 128T MIB Database tree is structured as shown in the following diagram:



Note that the SNMP MIB objects only support read-only access (eg: GET, GETNEXT). To configure or alter the system it is necessary to use user-interfaces like PCLI or Web GUI or configuration APIs (eg: Netconf).

The MIBs are described individually below.

T128-MIB.mib

The parent T128-MIB defines the overall structure of the MIB and the products supported. Only the t128NetworkingPlatform currently exists however future MIBs may expand this to other 128T products.

T128-ALARM.mib

The T128-ALARM-MIB defines objects that describes the state of 128T alarms. The MIB also contains definitions for the SNMP trap notifications sent out by the system on event of an alarm being added or cleared.

There are two tables for viewing the current alarm state, t128AlarmTable and t128ShelvedAlarmTable. Each row in this table represents a unique alarm that is active in the system. The t128ShelvedAlarmTable represents alarms that are shelved (for instance if the router is marked as being in "maintenance-mode"). The t128AlarmTable represents current alarms that are active and not shelved. The key for each row is an instance identifier string, either t128AlarmTableInstanceId or t128ShelvedAlarmTableInstanceId depending on which table the alarm resides in. This string can be used to indicate a specific instance of an alarm, for example "Interface operationally down on eth0", which would be a different instance than a "Interface operationally down on dpdk1" alarm. This instance identifier should be considered an opaque string, ie: the contents have no meaning other than that for each alarm this string is garaunteed to be unique. Note that these opaque instance identifiers are currently implemented as hex strings however the implementation may change from release to release so the exact content should not be relied on.

The t128AlarmNotification definition defines the alarm trap sent out by the system and contains the relevant objects to describe the alarm (eg: source node and router, severity, alarm message, etc). The trap will also contain a unique t128AlarmInstanceId that can be used to correlate the alarm with the t128AlarmTable and t128ShelvedAlarmTable instance identifiers. If an alarm moves from a shelved to unshelved state or vice versa the instance ID will remain the same.

T128-METRICS.mib

The 128T-METRICS-MIB provides SNMP access to the expansive set of metrics provided by the 128T product which are exposed via the t128MetricsTable.

Each row of the metrics table is keyed by two objects, the t128MetricAlias and the t128MetricIndex. The t128MetricAlias is an arbitrary string of up to 64 characters long that represents a description of the metric. When read from the table this alias also includes a metric instance suffix which in the most common case will be "_0". For example the default CPU utlization metric has an alias "cpuUtilization" will be returned as "cpuUtilization_0". In addition to the alias the t128MetricIndex key represents a unique series of a metric.

The values contained in each row are the t128MetricValue and t128MetricContributors objects. The t128MetricValue is simply the current value of the metric. The unique parameters that identify each data series are referred to as "contributors". For the CPU metric the contributors are router, node and core and the t128MetricContributors object will contain a string representation of these values so that each row can be correlated with the correct metric data series.

Below is an example of an SNMP query of the t128MetricsTable showing the 4 rows returned for each CPU utilization data series:

```
snmptable -Cl -Ci -mALL -v2c -c public 172.18.1.55 T128-METRICS-
MIB::t128MetricsTable

SNMP table: T128-METRICS-MIB::t128MetricsTable

index                   t128MetricContributors    t128MetricValue
"cpuUtilization_0".0    Fabric128.test1.3         5
"cpuUtilization_0".1    Fabric128.test1.2         2
"cpuUtilization_0".2    Fabric128.test1.1         100
"cpuUtilization_0".3    Fabric128.test1.0         4
```

Note that in this example the router name is "Fabric128", the node is called "test1" and the CPU cores are 0, 1, 2 and 3. This can be clearly seen in the t128MetricContributors column, eg: "Fabric128.test1.0".

The metrics table can be configured to contain any of the 128T metrics that can be exposed via SNMP however this is an advanced feature and is outside the scope of this document. By default a small set of the most useful metrics are configured in the t128MetricsTable.

T128-SYSTEM.mib

The T128-SYSTEM-MIB contains objects to describe the running state of the local 128T system.

The local node and router name, role and asset-id objects (t128RouterName, t128NodeName, t128NodeRole and t128AssetId) can be used to identify the system. The values of these parameters will match those in the 128T system configuration.

There are several processes running on the system that comprise the 128T product. The table t128ProcessTable provides details about these processes, for instance whether they are running or, if applicable, are assuming a leadership role. The key for the table rows is the process name, for example "nodeMonitor".

---

1. https://en.wikipedia.org/wiki/Simple_Network_Management_Protocol ↩