Zabbix Advanced

Aula 03: Coleta Avançada com SNMP e MIBs

4Linux - Curso Avançado

Agenda do Dia

1. **Fundamentos do SNMP**
    - Arquitetura e versões (v1, v2c, v3)
2. **Trabalhando com MIBs**
    - Estrutura OID, MIBs fundamentais e especializadas
3. **Configuração no Zabbix**
    - Community strings, SNMPv3, Discovery
4. **Templates Especializados**
    - Cisco, HP, Dell, impressoras, UPS

Agenda do Dia (continuação)

5. **Troubleshooting SNMP**
    - Problemas comuns e ferramentas

PARTE 1

Fundamentos do Protocolo SNMP

Objetivos de Aprendizagem

Ao final desta aula, você será capaz de:

- ✅ Compreender SNMP e suas versões
- ✅ Trabalhar com MIBs de forma prática
- ✅ Configurar coleta SNMP no Zabbix
- ✅ Implementar discovery de interfaces
- ✅ Criar templates especializados
- ✅ Diagnosticar problemas SNMP
- ✅ Otimizar performance de coleta

Recap Aula 02

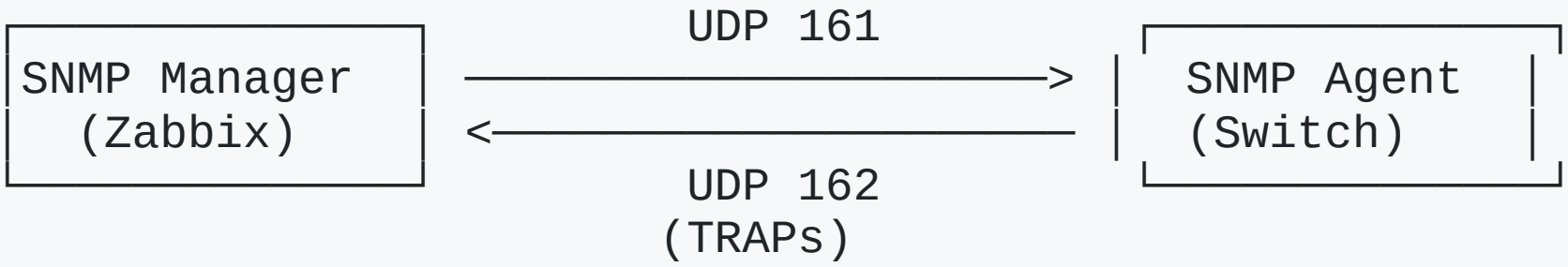
O que vimos:

- Métodos de coleta (Agent, SNMP, HTTP)
- Agente Zabbix (UserParameters)
- SNMP básico (versões, community)
- HTTP Agent e APIs REST
- Auto-registro

Hoje: SNMP avançado e MIBs! 🚀

Arquitetura SNMP

Componentes:



3 Elementos:

- **SNMP Manager** → Coleta dados (Zabbix Server)
- **SNMP Agent** → Fornece dados (Switch, Router)
- **MIB** → Base de dados estruturada

Versões do SNMP

SNMPv1 (1988):

- ✅ Primeira versão, universal
- ❌ Sem criptografia (plain text)
- ❌ Contadores 32-bit
- ❌ Community string simples

SNMPv2c (1993):

- ✅ Contadores 64-bit (Counter64)
- ✅ **GET-BULK** (70% mais eficiente!)
- ✅ Melhores códigos de erro
- ❌ Ainda sem criptografia

SNMPv3: Segurança

Níveis de Segurança:

```
noAuthNoPriv  → Sem autenticação, sem criptografia
authNoPriv    → Com autenticação, sem criptografia
authPriv      → Com autenticação E criptografia ✅ USAR!
```

Algoritmos:

- **Autenticação:** MD5, SHA-1, SHA-256, SHA-512
- **Criptografia:** DES, 3DES, AES-128, AES-192, AES-256

Produção: Sempre SNMPv3 authPriv! 🔒

Operações SNMP

Operação	Porta	Direção	Quando Usar
GET	161	Manager → Agent	1 OID específico
GET-NEXT	161	Manager → Agent	Próximo OID
GET-BULK	161	Manager → Agent	Múltiplos OIDs (⚡ **70% mais rápido**)
SET	161	Manager → Agent	Modificar valor
TRAP	162	Agent → Manager	Alerta imediato
INFORM	162	Agent → Manager	TRAP com ACK

GET vs GET-BULK: Performance

Cenário: Coletar 100 interfaces

Com GET:

```
snmpget ... ifDescr.1
snmpget ... ifDescr.2
snmpget ... ifDescr.3
...
# 100 requisições = 10 segundos
```

Com GET-BULK:

```
snmpbulkwalk ... ifDescr
# 2 requisições = 3 segundos
```

⚡ **70% mais rápido!**

Demonstração: Versões SNMP

```
# SNMPv1 - Simples, inseguro
snmpwalk -v1 -c public 192.168.1.1 \
  1.3.6.1.2.1.1.1.0

# SNMPv2c - GET-BULK eficiente
snmpbulkwalk -v2c -c public 192.168.1.1 \
  1.3.6.1.2.1.2.2.1.2

# SNMPv3 - Seguro (auth + criptografia)
snmpget -v3 -u zabbix-user -l authPriv \
  -a SHA -A myauthpass \
  -x AES -X myprivpass \
  192.168.1.1 1.3.6.1.2.1.1.1.0
```

Vamos testar ao vivo!

TRAP vs Polling

Polling (GET) - Reativo:

```
[Zabbix pergunta a cada 60s]
"Está tudo ok?" → "Sim"
"Está tudo ok?" → "Sim"
"Está tudo ok?" → "Sim"
```

TRAP - Proativo:

```
[Switch avisa imediatamente]
"ALERTA! Interface eth0 caiu!" 🚨
```

TRAPs comuns:

- `linkDown` - Interface caiu
- `coldStart` - Reboot do equipamento
- `authenticationFailure` - Acesso não autorizado

PARTE 2

Trabalhando com MIBs

O que são MIBs?

MIB = Management Information Base

Base de dados hierárquica estruturada em **árvore**

Cada nó = **OID** (Object Identifier)

```
iso (1)
 └── org (3)
     └── dod (6)
         └── internet (1)
             ├── mgmt (2)
             │   └── mib-2 (1)
             │       ├── system (1)
             │       ├── interfaces (2)
             │       └── ip (4)
             └── private (4)
                 └── enterprises (1)
```

OID: Numérico vs Nome

OID Numérico:

```
1.3.6.1.2.1.1.1.0
```

OID Nome:

```
SNMPv2-MIB::sysDescr.0
```

São a mesma coisa!

Converter:

```
# Nome → Numérico
snmptranslate -On SNMPv2-MIB::sysDescr.0
# Output: .1.3.6.1.2.1.1.1.0
# Numérico → Nome
```

MIB-II (RFC1213): System Group

OIDs Fundamentais:

OID	Nome	Descrição	Exemplo
1.3.6.1.2.1.1.1.0	sysDescr	Descrição	"Cisco IOS 15.2"
1.3.6.1.2.1.1.3.0	sysUpTime	Uptime	1234567 (timeticks)
1.3.6.1.2.1.1.5.0	sysName	Hostname	"switch-core-01"
1.3.6.1.2.1.1.6.0	sysLocation	Local	"Datacenter Rack 42"

Teste:

```
snmpwalk -v2c -c public 192.168.1.1 1.3.6.1.2.1.1
```

Interface MIB (IF-MIB)

OIDs Mais Importantes:

OID	Nome	O que é	Uso
1.3.6.1.2.1.2.2.1.2.X	ifDescr	Nome interface	"eth0", "Gi0/1"
1.3.6.1.2.1.2.2.1.5.X	ifSpeed	Velocidade (bps)	1000000000 = 1Gbps
1.3.6.1.2.1.2.2.1.8.X	ifOperStatus	Status	1=up, 2=down
1.3.6.1.2.1.2.2.1.10.X	ifInOctets	Bytes IN (32-bit)	Tráfego entrada
1.3.6.1.2.1.2.2.1.16.X	ifOutOctets	Bytes OUT (32-bit)	Tráfego saída
1.3.6.1.2.1.31.1.1.1.6.X	ifHCInOctets	Bytes IN (64-bit)	Para links ≥1Gbps

X = Index da interface (1, 2, 3...)

Problema: Wrap de Contador 32-bit

Contador 32-bit:

- Máximo: 4.294.967.295 bytes ≈ 4GB
- Link 1Gbps transfere 4GB em **34 segundos**
- Contador reseta (wrap) → Gráfico mostra pico negativo ❌

Solução: Contador 64-bit

```
ifInOctets     (32-bit) → Links <100Mbps
ifHCInOctets   (64-bit) → Links ≥100Mbps ✅
```

Regra:

- Link < 100Mbps → Use Counter32
- Link ≥ 100Mbps → Use Counter64 (HC = High Capacity)

🔬 Laboratório Prático 1

Objetivo: Explorar MIBs fundamentais

Tarefas (30 min):

1. Consultar System Group:

```
snmpwalk -v2c -c public <ip> 1.3.6.1.2.1.1
```

2. Listar todas interfaces:

```
snmpwalk -v2c -c public <ip> 1.3.6.1.2.1.2.2.1.2
```

3. Ver status operacional:

```
snmpwalk -v2c -c public <ip> 1.3.6.1.2.1.2.2.1.8
```

4. Calcular tráfego de interface 2 (coleta 2x com 60s de intervalo)

MIBs Especializadas: Cisco

CISCO-PROCESS-MIB (CPU):

```
# CPU utilization (5min average)
snmpwalk -v2c -c public cisco-device \
  1.3.6.1.4.1.9.9.109.1.1.1.1.7
```

CISCO-ENVMON-MIB (Temperatura/Fan):

```
# Temperature sensors
snmpwalk -v2c -c public cisco-device \
  1.3.6.1.4.1.9.9.13.1.3.1.3

# Fan status
snmpwalk -v2c -c public cisco-device \
  1.3.6.1.4.1.9.9.13.1.4.1.3
```

MIBs Especializadas: HP e Dell

HP ProLiant:

```
# System Health Status
snmpget -v2c -c public hp-server \
  1.3.6.1.4.1.232.6.2.6.1.0

# Drive Array Status
snmpwalk -v2c -c public hp-server \
  1.3.6.1.4.1.232.3.2.3.1.1.4
```

Dell PowerEdge:

```
# Overall System Status
snmpget -v2c -c public dell-server \
  1.3.6.1.4.1.674.10892.1.200.10.1.2.1

# Temperature Probes
snmpwalk -v2c -c public dell-server \
  1.3.6.1.4.1.674.10892.1.700.20.1.6
```

MIBs: Impressoras e UPS

Printer-MIB (Impressoras HP):

```
# Status da impressora
1.3.6.1.2.1.25.3.2.1.5.1

# Nível de toner (CMYK)
1.3.6.1.2.1.43.11.1.1.9.1.1  # Black
1.3.6.1.2.1.43.11.1.1.9.1.2  # Cyan
1.3.6.1.2.1.43.11.1.1.9.1.3  # Magenta
1.3.6.1.2.1.43.11.1.1.9.1.4  # Yellow

# Páginas impressas
1.3.6.1.2.1.43.10.2.1.4.1.1
```

MIBs: UPS APC

UPS-MIB (APC):

```
# Status da bateria
1.3.6.1.4.1.318.1.1.1.2.1.1.0

# Capacidade da bateria (%)
1.3.6.1.4.1.318.1.1.1.2.2.1.0

# Voltagem da bateria
1.3.6.1.4.1.318.1.1.1.2.2.8.0

# Temperatura
1.3.6.1.4.1.318.1.1.1.2.2.2.0

# Carga (%)
1.3.6.1.4.1.318.1.1.1.4.2.3.0
```

PARTE 3

Configuração Avançada no Zabbix

Community Strings

SNMPv1/v2c usa community strings:

Tipos:

- **RO (Read-Only):** Apenas leitura
- **RW (Read-Write):** Leitura + escrita

Configuração Linux:

```
sudo vim /etc/snmp/snmpd.conf

# Read-only para rede local
rocommunity public default
rocommunity monitoring 192.168.1.0/24

# Read-write para IP específico
rwcommunity private 192.168.1.100

sudo systemctl restart snmpd
```

SNMPv3: Configuração Segura

Criar usuário SNMPv3:

```
# Método 1: Tool automática
sudo net-snmp-create-v3-user -ro \
  -A SHA -a "authentication_password" \
  -X AES -x "privacy_password" \
  zabbix_user

# Método 2: Manual no snmpd.conf
sudo vim /etc/snmp/snmpd.conf
createUser zabbix_user SHA "auth_pass" AES "priv_pass"
rouser zabbix_user

sudo systemctl restart snmpd
```

Testar:

```
snmpget -v3 -u zabbix_user -l authPriv \
  -a SHA -A auth_pass -x AES -X priv_pass \
  localhost 1.3.6.1.2.1.1.1.0
```

Discovery de Interfaces (LLD)

Low-Level Discovery via SNMP:

Como funciona:

1. Zabbix consulta OID de discovery (ex: ifDescr)
2. Retorna lista de interfaces: {#IFNAME}, {#IFINDEX}
3. Cria automaticamente:
    - Items (tráfego IN/OUT por interface)
    - Triggers (interface down)
    - Graphs (gráfico de tráfego)

OID de Discovery:

```
1.3.6.1.2.1.2.2.1.2 → ifDescr (nome das interfaces)
```

Configurar Discovery no Zabbix

Passo a passo:

1. **Configuration → Hosts → Discovery → Create discovery rule**

```
Name: Network interface discovery
Type: SNMP agent
Key: net.if.discovery
SNMP OID: walk[1.3.6.1.2.1.2.2.1.2]
Update interval: 1h
```

2. **Filters (opcional):**

```
{#IFNAME} matches regex ^(eth|ens|Gi).*
{#IFOPERSTATUS} matches 1  (apenas interfaces UP)
```

3. **Item prototypes, Trigger prototypes, Graph prototypes**
    - Criados automaticamente pelo template

🔬 Laboratório Prático 2

Objetivo: Discovery de interfaces SNMP

Tarefas (30 min):

1. Criar host com interface SNMP
2. Aplicar template "Template Net Cisco Generic SNMPv2"
3. Aguardar discovery (ou executar manualmente)
4. Verificar:
    - Configuration → Hosts → Discovery (regra executou?)
    - Configuration → Hosts → Items (quantos items?)
5. Customizar filtros:
    - Incluir apenas interfaces "up"
    - Excluir loopback

Otimização de Performance

Timeout e Retry:

```
# /etc/zabbix/zabbix_server.conf
Timeout=10                      # Timeout global
StartSNMPTrapper=5             # Processos SNMP

# Por item (Zabbix frontend)
Update interval: 60s (crítico), 300s (informativo)
Timeout: 10s
```

Bulk Operations:

```
❌ GET individual:  100 interfaces = 100 requisições (10s)
✅ GET-BULK:        100 interfaces = 2 requisições (3s)
```

Intervalo de Coleta:

PARTE 4

Templates Especializados

Template Cisco Avançado

Componentes:

Itens:

- CPU utilization (5min avg)
- Memory utilization
- Device temperature
- Fan status

Triggers:

- High CPU (>80% por 5min) → Warning
- High temperature (>60°C) → High
- Fan problem (status ≠ normal) → High

Template Cisco: Items

```
<!-- CPU específico Cisco -->
<item>
  <name>Cisco CPU utilization</name>
  <type>SNMP_AGENT</type>
  <snmp_oid>1.3.6.1.4.1.9.9.109.1.1.1.1.7.1</snmp_oid>
  <key>cisco.cpu.util</key>
  <units>%</units>
</item>

<!-- Temperatura -->
<item>
  <name>Cisco Device Temperature</name>
  <type>SNMP_AGENT</type>
  <snmp_oid>1.3.6.1.4.1.9.9.13.1.3.1.3.1</snmp_oid>
  <key>cisco.temperature</key>
  <units>°C</units>
</item>
```

Template Cisco: Triggers

```
<!-- Trigger CPU -->
<trigger>
  <expression>
    avg(/Template Net Cisco Advanced SNMP/cisco.cpu.util,5m)>80
  </expression>
  <name>Cisco: High CPU utilization</name>
  <priority>WARNING</priority>
</trigger>

<!-- Trigger Temperatura -->
<trigger>
  <expression>
    last(/Template Net Cisco Advanced SNMP/cisco.temperature)>60
  </expression>
  <name>Cisco: High temperature</name>
  <priority>HIGH</priority>
</trigger>
```

Demonstração: Importar Template

Ao vivo:

1. Configuration → Templates → Import
2. Selecionar arquivo XML (fornecido)
3. Import
4. Analisar:
    - Items criados
    - Triggers criados
    - Value maps criados
5. Aplicar em switch Cisco
6. Validar coleta de dados

Deliverables da Atividade

Cada grupo deve:

1. Listar **5 OIDs importantes**
2. Criar **3 itens** no Zabbix
3. Criar **2 triggers**
4. Criar **1 value mapping** (se aplicável)
5. Testar coleta de dados
6. **Apresentar** (5 min/grupo):
    - Quais OIDs escolheram?
    - Por que são importantes?
    - Demonstração de Latest Data

Tempo total: 30 min trabalho + 15 min apresentações

PARTE 5

Troubleshooting SNMP

5 Problemas Comuns

1. Timeout em consultas

→ Aumentar timeout, usar GET-BULK, verificar firewall

2. Community string incorreta

→ "No response" → Verificar config do dispositivo

3. SNMPv3 authentication failure

→ Validar senha, algoritmo (SHA vs MD5)

4. OID not supported

→ MIB não existe → snmpwalk completo, consultar docs

5. Contadores resetando

→ Reboot ou wrap → Usar Counter64

Problema 1: Timeout

Sintoma:

```
snmpwalk -v2c -c public 192.168.1.1 1.3.6.1.2.1.2
Timeout: No Response from 192.168.1.1
```

Diagnóstico:

1. Ping funciona? `ping 192.168.1.1`
2. Porta aberta? `nmap -sU -p 161 192.168.1.1`
3. Community correta? Testar com sysDescr

Solução:

```
# Aumentar timeout
snmpwalk -v2c -c public -t 10 192.168.1.1 1.3.6.1.2.1.2

# Usar GET-BULK (mais rápido)
snmpbulkwalk -v2c -c public 192.168.1.1 1.3.6.1.2.1.2
```

Problema 2: Community Incorreta

Sintoma:

```
Timeout: No Response from 192.168.1.1
(mas ping funciona)
```

Diagnóstico:

```
# Testar community strings comuns
snmpget -v2c -c public 192.168.1.1 1.3.6.1.2.1.1.1.0
snmpget -v2c -c private 192.168.1.1 1.3.6.1.2.1.1.1.0
snmpget -v2c -c monitoring 192.168.1.1 1.3.6.1.2.1.1.1.0
```

Solução:

- Verificar config do switch: `show snmp community`
- Reconfigurar: `snmp-server community monitoring ro`

Problema 3: SNMPv3 Auth Failure

Sintoma:

```
Authentication failure (incorrect password, community or key)
```

Diagnóstico:

```
# Verificar usuário existe no dispositivo
snmpusm -v3 -u zabbix_user 192.168.1.1

# Testar diferentes algoritmos
snmpget -v3 -u user -l authPriv \
  -a MD5 -A pass1 -x DES -X pass2 ...  # Algoritmo 1

snmpget -v3 -u user -l authPriv \
  -a SHA -A pass1 -x AES -X pass2 ...  # Algoritmo 2
```

Solução:

- Validar senha correta
- Verificar algoritmo suportado pelo dispositivo

Problema 4: OID Not Found

Sintoma:

```
No Such Object available on this agent at this OID
```

Diagnóstico:

```
# Fazer snmpwalk completo (ver o que existe)
snmpwalk -v2c -c public 192.168.1.1 1.3.6

# Converter OID nome → numérico
snmptranslate -On IF-MIB::ifDescr.1
```

Solução:

- Consultar documentação do fabricante
- Verificar se MIB está carregada
- Usar OID alternativo (ex: ifDescr vs ifName)

Problema 5: Contadores Resetando

Sintoma:

- Gráfico mostra picos negativos
- Tráfego "zera" periodicamente

Causa:

- Reboot do dispositivo
- Overflow de contador 32-bit (wrap)

Solução:

```
❌ ifInOctets     (32-bit)  → Wrap em 34s (link 1Gbps)
✅ ifHCInOctets   (64-bit)  → Sem wrap

Preprocessing no Zabbix:
- Change per second (delta)
- Simple change
```

Ferramentas de Debugging

snmptranslate:

```
# OID → Nome
snmptranslate 1.3.6.1.2.1.1.1.0
# Output: SNMPv2-MIB::sysDescr.0

# Nome → OID
snmptranslate -On SNMPv2-MIB::sysDescr.0
# Output: .1.3.6.1.2.1.1.1.0

# Informações detalhadas
snmptranslate -Td SNMPv2-MIB::sysDescr.0
```

tcpdump e Wireshark

Capturar tráfego SNMP:

```
# Captura simples
sudo tcpdump -i any port 161 -w snmp.pcap

# Captura com análise em tempo real
sudo tcpdump -i any -s 0 -A port 161 and host 192.168.1.10
```

Filtros Wireshark:

```
snmp                          # Todo tráfego SNMP
snmp.version == 2             # Apenas SNMPv2
snmp.pdu_type == 0            # Apenas GET
snmp.pdu_type == 5            # Apenas GET-BULK
snmp.name contains "ifDescr"  # OID específico
```

ENCERRAMENTO

Recap dos Principais Conceitos

✅ **SNMP:** v1 (inseguro), v2c (GET-BULK), v3 (seguro)
✅ **MIBs:** Estrutura OID, RFC1213, IF-MIB, fabricantes
✅ **Performance:** GET-BULK 70% mais rápido
✅ **Discovery:** LLD automático de interfaces
✅ **Templates:** Especializados por fabricante
✅ **Counter64:** Para links ≥100Mbps
✅ **Troubleshooting:** 5 problemas + ferramentas
✅ **SNMPv3:** Sempre authPriv em produção 🔒

Comparação Final

Versão	Segurança	Performance	Quando Usar
SNMPv1	❌ Nenhuma	🔸 Média	Lab/legado
SNMPv2c	❌ Nenhuma	⚡ Alta (GET-BULK)	Rede interna
SNMPv3	✅ Auth+Crypt	⚡ Alta	**Produção!**

Regra de ouro:

- Lab/teste → SNMPv2c
- Produção → **SNMPv3 authPriv** 🔒

Recursos Úteis

📚 **RFCs:**

- RFC 1157 (SNMPv1): https://www.ietf.org/rfc/rfc1157.txt
- RFC 1213 (MIB-II): https://www.ietf.org/rfc/rfc1213.txt
- RFC 3410-3418 (SNMPv3): https://www.ietf.org/rfc/rfc3410.txt

🔍 **Ferramentas:**

- OID Repository: http://www.oid-info.com/
- MIB Browser: http://www.ireasoning.com/mibbrowser.shtml

📖 **Docs:**

- Cisco MIBs: https://www.cisco.com/c/en/us/support/docs/ip/simple-network-management-protocol-snmp/17282-snmp-mibs.html

Tabela de OIDs Essenciais

System (1.3.6.1.2.1.1):

- .1.0 → sysDescr
- .3.0 → sysUpTime
- .5.0 → sysName

Interfaces (1.3.6.1.2.1.2.2.1):

- .2.X → ifDescr
- .8.X → ifOperStatus
- .10.X → ifInOctets (32-bit)
- .16.X → ifOutOctets (32-bit)

High Capacity (1.3.6.1.2.1.31.1.1.1):

- .6.X → ifHCInOctets (64-bit) ✅
- .10.X → ifHCOutOctets (64-bit) ✅

Obrigado!

Até a próxima aula! 🚀