Less-FS spart auf Linux-Dateisystemen Platz | Neu: Small Business Server 2011 | Yubikey sichert Wordpress-Login

01 2011 Jan. – Feb.

ADMIN

Netzwerk & Security

Auf CD:

MONITORING

SERVER UND DIENSTE UNTER KONTROLLE

IPv6 VOR DEM START
Tipps zur Konfiguration und Migration

HELPDESKS
Ticket-Systeme im Überblick
Request Tracker im Detail

Nagios-IPMI
Server-Hardware direkt überwacht

Icinga
Konfiguration im LDAP

SSH Guard
Secure Shell und andere Dienste absichern

Splunk
Business Process Monitoring

Linux-Cluster
Mit Open AIS selbstgemacht

iLiveX
Linux-Desktop auf dem iPad

Openstack
Die offene API für die Cloud

ADMIN - das Magazin für alle IT-Administratoren von Linux und Windows.
Mehr Informationen unter
http://www.admin-magazin.de

www.admin-magazin.de

4 196360 509805 01

beerkoff, Fotolia

Serverhardware mit dem IPMI-Plugin für Nagios überwachen

Lebenszeichen

Nagios und Icinga können nicht nur Serverdienste überwachen, sondern mit den richtigen Helfern auch die darunterliegende Serverhardware. Der Autor des dafür maßgeschneiderten IPMI-Plugin gibt einen Überblick über seine Software. Werner Fischer

Systeme wie Nagios oder Icinga haben sich in den letzten Jahren beim Monitoring von Softwarediensten bewährt. Die Überwachung von Serverhardware war aber bisher nur aufwändig mit Hersteller-spezifischen Plugins möglich. Das neue IPMI-Plugin (Version 2) erlaubt die einfache Überwachung selbst heterogener Serverlandschaften. Es überwacht alle IPMI-Hardwaresensoren, etwa Temperaturen, Lüfter-Drehzahlen, Netzteil-Status und viele andere mehr.

IPMI (Intelligent Platform Management Interface) wurde bereits 1998 als Hersteller-übergreifender Standard fürs Servermanagement von Intel, HP, NEC und Dell verabschiedet. Die aktuelle Ausgabe IPMI 2.0 stammt aus dem Jahr 2004. Sie wird praktisch von allen modernen Serversystemen unterstützt. Einstiegsserver unterstützen IPMI oft optional, etwa über Erweiterungskarten oder spezielle Mainboard-Varianten. Bei den restlichen Servern gehört IPMI aber meist zur Serienausstattung ([1], [2], [3]).

Das Kernstück von IPMI bildet der so genannte Baseboard Management Controller (BMC). Er kommuniziert einerseits über das Netzwerk oder einen lokalen Systembus mit Userland-Programmen, andererseits ist er mit zahlreichen Hardwaresensoren im Server verbunden. Für die Kommunikation übers Netzwerk bekommt der BMC eine eigene IP-Adresse. Sobald der Server an das Stromnetz angeschlossen ist, bootet der BMC automatisch – unabhängig davon, ob der Server selbst läuft.

Die breite IPMI-Unterstützung im Serverumfeld ist die ideale Voraussetzung für ein Nagios/Icinga-Plugin zur einfachen und einheitlichen Überwachung von Serverhardware. Das IPMI-Sensor-Monitoring-Plugin des Autors ist im Oktober 2009 in einer ersten Version erschienen. Im Hintergrund nutzt es Ipmitool zur Abfrage der IPMI-Sensoren. Seit Kurzem gibt es das Plugin in der Version 2.0. Im Hintergrund läuft nun »ipmimonitoring« von Free IPMI. Die Umstellung von Ipmitool auf Free IPMI war notwendig, um neben den analogen Sensoren (Threshold Sensors) auch digitale Sensoren (Discrete Sensors) zuverlässig zu überwachen. Free IPMI wird mittlerweile von immer mehr Linux-Distributionen direkt bereitgestellt, etwa von RHEL und Centos ab Version 5.2, Ubuntu ab Version 10.04 oder Debian Squeeze [4].

Die Sensor-Klassen Threshold und Discrete

Die Unterscheidung in die zwei Sensor-Klassen Threshold und Discrete ist in der IPMI-Spezifikation standardisiert. Abbildung 1 zeigt einen Threshold Sensor (Fan 1). Ein solcher Sensor liefert einen analogen Messwert (in diesem Beispiel 5719 U/min). Zusätzlich stellt er auch eine Zustandsinformation bereit (hier »ok«). Die generiert der Sensor durch den Vergleich des analogen Messwerts mit den vordefinierten Grenzwerten. Bei diesem Lüfter sind keine Obergrenzen definiert,

```
[root@testserver ~]# ipmitool sdr get "Fan 1"
Sensor ID              : Fan 1 (0x50)
 Entity ID             : 29.1 (Fan Device)
 Sensor Type (Analog)  : Fan
 Sensor Reading        : 5719 (+/- 0) RPM
 Status                : ok
 Nominal Reading       : 6708.000
 Normal Minimum        : 2451.000
 Normal Maximum        : 10965.000
 Lower critical        : 1720.000
 Lower non-critical    : 1978.000
 Positive Hysteresis   : 86.000
 Negative Hysteresis   : 86.000
 Minimum sensor range  : Unspecified
 Maximum sensor range  : Unspecified
 Event Message Control : Per-threshold
 Readable Thresholds   : lcr lnc
 Settable Thresholds   : lcr lnc
 Threshold Read Mask   : lcr lnc
 Assertion Events      :
 Assertions Enabled    : lnc- lcr-
 Deassertions Enabled  : lnc- lcr-
[...]
```

Abbildung 1: Daten eines Threshold-Sensors, der analoge Messwerte liefert.

sondern nur zwei Untergrenzen LNC (Lower non critical) und LCR (Lower critical) bei 1978 und 1720 U/min. Hier offenbart sich ein weiterer Vorteil von IPMI: Die Grenzwerte sind bereits vom Serverhersteller definiert. Das erspart die manuelle Konfiguration von Grenzwerten in Nagios oder Icinga.

Abbildung 2 zeigt einen Discrete Sensor (PS1 Status). Dieser Sensor stellt Statusinformationen des ersten Netzteils bereit. Er liefert dabei aber keinen analogen Messwert. Stattdessen zeigt er, welcher seiner vier möglichen Zustände eingetreten ist. Es können dabei auch mehrere Zustände gleichzeitig aktiv sein. Das ist auch hier der Fall. Die beiden Zustände »Presence detected« und »Power Supply AC lost« sind aktiv. Ipmitool generiert hier aber leider bei einer Abfrage »ipmitool sdr elist all« keine Warnung.

Im Gegensatz dazu verfügt »ipmimonitoring« von Free IPMI über eine genaue Zuordnung, welche diskreten Zustände als Okay (Nominal), Warning oder Critical interpretiert werden sollen. Diese Abstufung entspricht den gleichnamigen Zuständen bei Nagios/Icinga. Die Standardzuordnung von »ipmimonitoring« lässt sich über die Konfigurationsdatei »/etc/ipmi_monitoring_sensors.conf« anpassen (Listing 1).

Sensoren abfragen

Die Abfrage der Hardwaresensoren ist grundsätzlich lokal oder übers Netzwerk möglich. Der lokale Zugriff auf den Server über ein IPMI-System-Interface erfordert Root-Rechte, eine einfache Freigabe des »ipmimonitoring«-Tools per »sudo« genügt aber. Diese Abfrageart ist für den Monitoringserver selbst beziehungsweise für Hosts sinnvoll, die bereits über NRPE abgefragt werden.

Der Remote-Zugang erfordert eine IP-Adresse für den IPMI-BMC, einen IPMI-Benutzernamen und ein Passwort. Der verwendete IPMI-Benutzer sollte dem IPMI-Channel-Privilege-Level »User« zugeordnet sein. Wenn ein Angreifer die Zugangsdaten mitschneidet, kann er den Server mit diesen Rechten per IPMI weder neu starten noch ausschalten. Mit dem IPMI-Channel-Privilege-Level »Administrator« besteht diese Gefahr.

Der große Vorteil der Abfrage übers Netzwerk liegt in der Unabhängigkeit vom eingesetzten Betriebssystem am Server. Egal ob auf dem Server Linux, Windows oder VMware läuft, dank der Abfrage übers Netzwerk muss kein Agent im Betriebssystem installiert sein.

Einbindung in Icinga

Das folgende Beispiel zeigt die IPMI-Überwachung eines Servers übers Netzwerk. Auf dem Server ist dazu ein IPMI-Benutzer mit User-Rechten eingerichtet. Nun geht es darum, das IPMI-Plugin in Icinga einzubinden. Die Konfiguration ist bei einem Nagios-System übrigens identisch. Als Voraussetzungen erfordert das Plugin die Bash-Shell, das Free-IPMI-Paket und Awk.

Das Plugin gibt es auf den Webseiten von Thomas Krenn zum Download [5]. Es wird nach dem Download einfach im Standard-Plugin-Ordner abgelegt. Eine Definition des Kommandos in der »commands.cfg« macht das IPMI-Plugin danach für die einzelnen Host- und Service-Definitionen nutzbar (Listing 2). Mit der Custom-Object-Variablen »_ipmi_ip« wird danach die jeweilige IP-Adresse des IPMI-BMC in den bestehenden Hostdefinitionen der Server ergänzt (Listing 3). Die abschließende Servicedefinition erfordert als Parameter nur mehr den Pfad zur Free-IPMI-Konfigurationsdatei, die IPMI-Benutzernamen, -Passwort und Channel-Privilege-Level enthält (Listing 4 und 5). Dank dieser Konfigurationsdatei müssen keine IPMI-Passwörter in der Icinga-Konfiguration abgelegt werden.

Da kein Passwort als Parameter an »ipmimonitoring« übergeben wird, tauchen keine Zugangsdaten in der Prozessliste auf. Zudem erlaubt diese Vorgehensweise die einfache Konfiguration weiterer Parameter von »ipmimonitoring« ohne Anpassungen am Plugin. Die Konfigurationsdatei sollte aber aus Sicherheitsgründen nur für den User »icinga« lesbar sein.

Icinga überwacht nun mit dem Plugin alle IPMI-Sensoren des jeweiligen Servers. Bei

```
[root@testserver ~]# ipmitool sdr get "PS1 Status"
Sensor ID              : PS1 Status (0x70)
 Entity ID             : 10.1 (Power Supply)
 Sensor Type (Discrete): Power Supply
 States Asserted       : Power Supply
                         [Presence detected]
                         [Power Supply AC lost]
 Assertion Events      : Power Supply
                         [Presence detected]
                         [Power Supply AC lost]
 Assertions Enabled    : Power Supply
                         [Presence detected]
                         [Failure detected]
                         [Predictive failure]
                         [Power Supply AC lost]
[...]
 Deassertions Enabled  : Power Supply
[...]
```

Abbildung 2: Daten eines Discrete-Sensors, der einzelne Zustände annehmen kann.

Listing 1: »ipmi_monitoring_sensors.conf«

```
01 # IPMI_Power_Supply
02 # IPMI_Power_Supply_Presence_Detected
   Nominal
03 # IPMI_Power_Supply_Power_Supply_Failure_Detected
   Critical
04 # IPMI_Power_Supply_Predictive_Failure
   Critical
05 # IPMI_Power_Supply_Power_Supply_Input_Lost_AC_DC
   Critical
```

Listing 2: Kommando-Definition

```
01 define command {
02   command_name  check_ipmi_sensor
03   command_line  $USER1$/check_ipmi_sensor -H $_
   HOSTIPMI_IP$ -f $ARG1$
04   }
```

Listing 3: Host-Definition

```
01 define host{
02   use           linux-server
03   host_name     centos4
04   alias         centos4
05   address       192.168.1.151
06   _ipmi_ip      192.168.1.211
07 }
```

Listing 4: Service-Definition

```
01 define service{
02   use                  generic-service
03   host_name            centos4
04   service_description  IPMI
05   check_command        check_ipmi_sensor!/etc/
   ipmi-config/ipmi.cfg
06   }
```

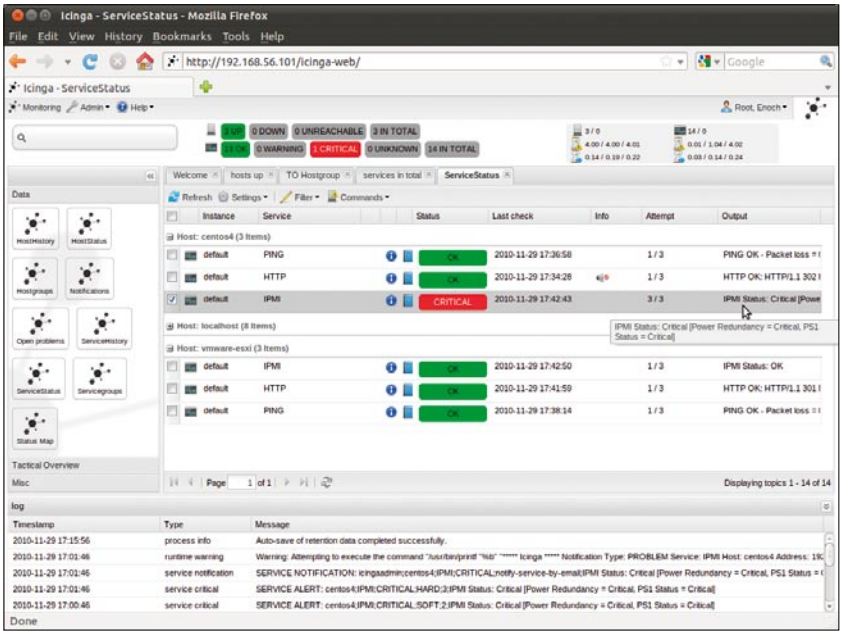

Abbildung 3: **Icinga zeigt den kritischen Status des IPMI-Service rot an.**

Fehlern schlägt Icinga sofort Alarm: Probleme mit der Stromversorgung führen umgehend zum Status Critical (Abbildung 3). Der Output »IPMI Status: Critical [Power Redundancy = Critical, PS1 Status = Critical]« deutet dabei auf ein Problem beim ersten Netzteil hin.

Wie es in den Nagios Plugin Development Guidelines vorgeschrieben ist, unterstützt das Plugin drei zusätzliche Verbosity-Level. Der erste Level sorgt für eine ausführlichere Ausgabe – in diesem Fall wäre dies: »IPMI Status: Critical [Power Redundancy = Critical ('Redundancy Lost' 'Non-redundant:Sufficient Resources from Redundant'), PS1 Status = Critical ('Presence detected' 'Power Supply input lost (AC/DC)')]«.

Ausführliche Infos

Mit diesen Details ist sofort ersichtlich, dass die Stromzufuhr zum ersten Netzteil unterbrochen ist, das Netzteil selbst aber keinen Fehler meldet. Aufgrund der höheren Zeichenanzahl kann aber eine solche Meldung bei SMS-Benachrichtigungen unter Umständen abgeschnitten werden. Der Verbosity-Level 2 liefert darüber hinaus eine mehrzeilige Ausgabe.

Listing 5: IPMI-Benutzerdaten

```
01 username monitor
02 password ao5$snNc!
03 privilege-level user
```

Level 3 enthält schließlich ausführliche Debugging-Informationen, die im Fehlerfall wertvolle Hinweise zu möglichen Konfigurationsproblemen liefern.

Das IPMI-Plugin stellt für alle numerischen Messwerte auch Performancedaten bereit. Sie lassen sich über die bekannten Visualisierungs-Tools, etwa mit PNP4Nagios, in Graphen darstellen. Abbildung 4 zeigt beispielsweise den Anstieg der Stromaufnahme des Netzteils 2 von 0,5 Ampere auf knapp 1 Ampere nach dem Ausfall der Stromzufuhr von Netzteil 1 kurz vor 17:00 Uhr. Auch für alle anderen numerischen Messwerte wie Lüfterdrehzahlen, Temperaturen oder Spannungen erstellt PNP4Nagios Performancegraphen.

Für das Jahr 2011 plant das Icinga-Team die Einbindung der neuen Version 2 des Netways-Grapher. Damit lassen sich die erhobenen Performancedaten als dynamische Graphen darstellen.

Fazit

Die neue Version 2 des IPMI-Plugin überwacht zuverlässig alle IPMI-Sensoren, egal ob Threshold- oder Discrete-Sensoren. Bei Problemen mit der Hardware senden Icinga oder Nagios sofort Benachrichtigungen an die zuständigen Administratoren. Blieben früher einzelne Ausfälle von Lüftern oder Netzteilen so lange unerkannt, bis schließlich der ganze Server ausfiel, erlaubt die IPMI-Überwachung nun eine rasche Fehlerbehebung.

Die bereitgestellten Performancedaten bieten einen weiteren Mehrwert: War in der Vergangenheit etwa nur die Überwachung eines Temperatursensors pro Rack üblich, liefert das IPMI-Plugin nun Temperaturen jedes einzelnen Servers. Damit lassen sich selbst nur lokal auftretende Kühlprobleme rasch ausfindig machen und beheben.

Mit der Nutzung des IPMI-Plugin lässt sich also die Verfügbarkeit einer ganzen Serverlandschaft drastisch erhöhen. Einem Einsatz steht nichts im Wege – das IPMI-Plugin ist als Open-Source-Software unter der GPLv3 lizenziert. *(ofr)* ■

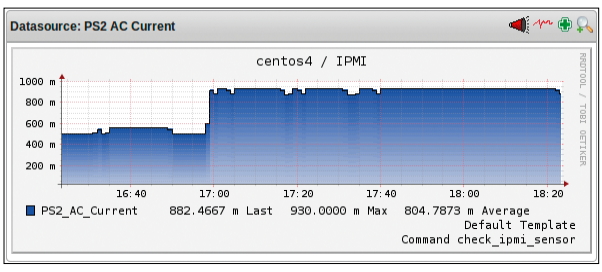

Abbildung 4: **Um 17:00 Uhr steigt plötzlich die Stromaufnahme des Servers auf beinahe 1 Ampere an.**

Infos

[1] Alles über IPMI: [http://www.linuxtechnicalreview.de/Vorschau/(show)/Themen/Monitoring/Alles-ueber-IPMI]

[2] Justin Penney, „Server überwachen und managen mit IPMI“: ADMIN 03/2010, S. 67

[3] IPMI-Grundlagen: [http://www.thomas-krenn.com/de/wiki/IPMI_Grundlagen]

[4] Versionsinformationen zu Free IPMI: [http://www.thomas-krenn.com/de/wiki/FreeIPMI]

[5] IPMI-Sensor-Monitoring-Plugin: [http://www.thomas-krenn.com/ipmi-plugin]

Der Autor

Werner Fischer ist seit 2005 Technology Specialist bei der Thomas-Krenn AG und Chefredakteur des Thomas-Krenn-Wiki. Seine Arbeitsschwerpunkte liegen in den Bereichen Hardwaremonitoring, Virtualisierung, I/O-Performance und Hochverfügbarkeit.