Chaves específicas para Windows

Chaves de item

A tabela a seguir apresenta detalhes das chaves de item que estão disponíveis somente no Zabbix Agent para Windows.

Chave			
Descrição	**Retorno**	**Parâmetros**	**Comentários**
eventlog[name,<regexp>,<severity>,<source>,<eventid>,<maxlines>,<mode>]			
Monitoramento de eventos em log.	Log	**name** - nome log de eventos **regexp** - expressão regular com o padrão desejado **severity** - expressão regular com a severidade desejada Este parâmetro aceita os seguintes valores: “*Information*”, “*Warning*”, “*Error*”, “*Critical*”, “*Verbose*” (desde o Zabbix 2.2.0 executando em Windows Vista ou superior) **source** - expressão regular descrevendo a fonte de identificação (o uso de expressão regular é suportado desde o Zabbix 2.2.0) **eventid** - expressão regular descrevendo os identificadores de evento **maxlines** - quantidade máxima de novas linhas por segundo que o Zabbix Agent enviará ao Zabbix Server ou Zabbix Proxy. Este parâmetro sobrescreve a definição 'MaxLinesPerSecond' que estiver definda no arquivo zabbix_agentd.win.conf **mode** - valores possíveis: *all* (padrão), *skip* - não processar os dados anteriores (afeta somente as linhas adicionadas após o início da coleta).	O item precisa ser configurado com o tipo Agente Zabbix (ativo). Exemplos: ⇒ eventlog[Application] ⇒ eventlog[Security,,"Failure Audit",,^(529|680)$] ⇒ eventlog[System,,"Warning\|Error"] ⇒ eventlog[System,,,,^1$] ⇒ eventlog[System,,,,@TWOSHORT] - aqui uma expressão regular customizada chamada **TWOSHORT** é referenciada (definida com o tipo **Resultado VERDADEIRO**, a expressão em sí seria **^1$\|^70$**). O parâmetro `mode` é suportado desde o Zabbix 2.0.0. “Windows Eventing 6.0” é suportado desde o Zabbix 2.2.0. Observe que selecionando um tipo de dado diferente de **Log** para este item ocasionará a perda do carimbo de hora, gravidade de evento e sua origem. Consulte também informações adicionais em monitoramento de log.
net.if.list			

Chave			
Descrição	**Retorno**	**Parâmetros**	**Comentários**
Lista de interfaces de rede (incluindo o tipo, status, endereço IPv4, descrição).	Texto		Suportado desde o Zabbix 1.8.1. Interfaces com caracteres Multi-byte no nome são suportadas desde o Zabbix 1.8.6. Interfaces desabilitadas não são listadas. Observe que habilitando/desabilitando alguns componentes poderá alterar a sua ordem na lista de interfaces do Windows. Algumas versões do Windows (por exemplo, Windows Server 2008) poderão requerer os últimos updates para suportar caracteres não ASCII nos nomes de interface.
perf_counter[counter,<interval>]			
Valor de qualquer contador do Windows.	Inteiro, numérico, string or texto (dependendo da requisição)	**counter** - caminho para o contador **interval** - últimos N segundos de armazenamento do valor. O parâmetro `interval` precisa estar entre 1 e 900 segundos (inclusive), o padrão é 1.	O monitor de performance pode ser utilizado também para obter a lista de contadores disponível. Antes da versão 1.6 este parâmetro retornaria corretamente somente se fosse seguido um padrão (algo como: \System\Threads). Pode não funcionar corretamente para contadores que necessitam de mais de uma mostra (como a utilização de CPU). Desde o Zabbix 1.6, o parâmetro `interval` é utilizado para se obter o valor médio para o último intervalo de N segundos. Maix detalhes em: Contadores de performance Windows.
proc_info[process,<attribute>,<type>]			

Chave			
Descrição	**Retorno**	**Parâmetros**	**Comentários**
Informações sobre processos específicos.	Numérico	**process** - nome do processo **attribute** - atributo de processo necessário **type** - unidade de representação (significativo quando existe mais de um processo com o mesmo nome)	O parâmetro `attributes` pode ser: *vmsize* (padrão) - tamanho da memória virtual do processo (em Kbytes) *wkset* - tamanho do pacote de trabalho (quantidade de memória física utilizada pelo processo) em Kbytes *pf* - quantidade de falhas de página *ktime* - tempo de kernel em milesegundos *utime* - tempo de usuário em milisegundos *io_read_b* - quantidade de bytes lidos pelo processo em operações de I/O *io_read_op* - quantidade de operações de leitura feitas pelo processo *io_write_b* - quantidade de bytes gravados pelo processo durante operações de I/O *io_write_op* - quantidade de operações de escrita feitas pelo processo *io_other_b* - quantidade de bytes transferidos pelo processo durante operações diversas *io_other_op* - quantidade de operações de I/O executadas pelo processo, que não sejam de leitura e gravação *gdiobj* - quantidade de objetos GDI utilizados pelo processo *userobj* - quantidade de objetos de usuário utilizados pelo processo Valores possíveis para o parâmetro `types`: *avg* (padrão) - valor médio de todos os processos **<process>** *min* - valor mínimo de todos os processos **<process>** *max* - valor máximo de todos os processos **<process>** *sum* - valor total de todos os processos **<process>** Exemplos: ⇒ proc_info[iexplore.exe,wkset,sum] - para obter o total de memória física por todos os processos do Internet Explorer ⇒ proc_info[iexplore.exe,pf,avg] - para obter a média de falhas de página do processo do Interent Explorer Observe que em um ambiente de 64-bit, uma versão do Zabbix Agent em 64-bit será necessária para o correto funcionamento. Nota: Os atributos *io_**, *gdiobj* e *userobj* estão disponíveis somente a partir do Windows 2000.

Chave			
Descrição	**Retorno**	**Parâmetros**	**Comentários**
service.discovery			
Lista os serviços do Windows. Usado pelo processo de autobusca.	Objeto JSON		Suportado desde o Zabbix 3.0.
service.info[service,<param>]			
Informação sobre um serviço.	Inteiro - com o parâmetro `param` definido como: *state*, *startup* String - com o parâmetro `param` definido como: *displayname*, *path*, *user* Texto - com o parâmetro `param` defiido como: *description* Especificamente para *state*: 0 - em execução, 1 - pausado, 2 - iniciação pendente, 3 - pausa pendente, 4 - retorno pendente, 5 - finalização pendente, 6 - finalizado, 7 - desconhecido, 255 - serviço desconhecido Especialmente para *startup*: 0 - automático, 1 - automático com atraso, 2 - manual, 3 - desabilitado, 4 - desconhecido	**service** - o nome real do serviço ou o nome de visualização como visto no snap MMC Services **param** - *state* (padrão), *displayname*, *path*, *user*, *startup* ou *description*	Exemplos: ⇒ service.info[SNMPTRAP] - estado do serviço SNMPTRAP ⇒ service.info[SNMP Trap] - estado do mesmo serviço, mas através do nome ⇒ service.info[EventLog,startup] - Eventos do serviço de EventLog com o tipo: inicialização Itens service.info[service,state] e service.info[service] retornarão a mesma informação. Note que apenas com o parâmetro `param` como *state* este item retornará o código de retorno para serviços inexistentes (255). Este item é suportado desde o Zabbix 3.0.0. Ele também pode ser utilizada a chave depreciada 'service_state[service]'.
services[<type>,<state>,<exclude>]			

Chave			
Descrição	**Retorno**	**Parâmetros**	**Comentários**
Lista de serviços.	0 - se vazia Texto - lista dos serviços separados por quebra de linha	**type** - *all* (padrão), *automatic*, *manual* ou *disabled* **state** - *all* (padrão), *stopped*, *started*, *start_pending*, *stop_pending*, *running*, *continue_pending*, *pause_pending* ou *paused* **exclude** - serviços a restringir no resultado. Os serviços a serem ignorados deverão estar entre aspas, separados por vírgulas e sem espaços.	Exemplos: ⇒ services[,started] - lista dos serviços iniciados ⇒ services[automatic, stopped] - ista dos serviços parados, mas que poderiam ser executados ⇒ services[automatic, stopped, "service1,service2,service3"] - lista dos serviços parados, que poderiam estar executando, excluindo os serviços: service1, service2 e service3 A restrição de serviços é suportado desde o Zabbix 1.8.1.
wmi.get[<namespace>,<query>]			
Executa uma consulta WMI e retorna o primeiro objeto encontrado.	Inteiro, numérico, string or texto (dependendo da requisição)	**namespace** - nome de espaço WMI **query** - consulta WMI retornando um objeto simples	Example: ⇒ wmi.get[root\cimv2,select status from Win32_DiskDrive where Name like '%PHYSICALDRIVE0%'] - retorna o status do primeiro disco físico. Esta chave é suportada desde o Zabbix 2.2.0.

Monitorando serviços do Windows

Este tutorial provê um passo-a-passo de como se configurar a monitoração de serviços do Windows. Partimos do princípio que o Zabbix Agent e o Zabbix Server estão configurados e operacionais.

Passo 1

Obter o nome do serviço.

Você pode obter o nome do serviço ao acessar o snap-in “MMC Services” e analisar as propriedades do serviço desejado. Na aba “Geral” existirá o campo 'Service name'. Este valor deverá er obtido para configurar o item de monitoração.

Por exemplo, se você precisa monitorar o serviço “workstation” então o nome do serviço seria: **lanmanworkstation**.

Passo 2

Configurar um item para monitorar o serviço.

A chave 'service.info[service,<param>]' recupera informações sobre um serviço. Dependendo do que você precisar, defina o parâmetro *param* para um destes valores: *displayname*, *state*, *path*, *user*, *startup* ou *description*. O valor padrão é *state* se *param* não for definido (service.info[service]).

O tipo do valor de retorno depende do que for definido no parâmetro *param*: inteiro para *state* e *startup*; string fara *displayname*, *path* e *user*; texto para *description*.

Exemplo:

- *Key*: service.info[lanmanworkstation]
- *Type of information*: Numeric (unsigned)
- *Show value*: select the *Windows service state* value mapping

Por padrão estão disponíveis no Zabbix dois mapeamentos de valores: *Windows service state* e *Windows service startup type* para mapear o valor numérico obtido para uma representação textual na interface web do Zabbix.

Descoberta de serviços do Windows

O processo de autobusca provê um caminho para a criação automática de itens, triggers, e graphs para diferentes entidades no mesmo computador. O Zabbix pode monitorar automaticamente os serviços Windows de sua máquina, sem que você precise saber exatamente o nome do serviço para criar os itens manualmente. Um filtro poderá ser utilizado para criar itens, triggers, e graphs apenas para os serviços de interesse.