Introducción a la monitorización

Volver al Indice

Introducción a la monitorización

Toda la interacción del usuario con Pandora FMS se realiza a través de la consola web. La consola permite acceso a través de un navegador sin necesidad de la instalación de aplicaciones pesadas, permitiendo la gestión desde cualquier equipo con un navegador.

La monitorización consiste en la ejecución de procesos sobre todo tipo de sistemas para recoger y almacenar información, realizar acciones y tomar decisiones en base a dichos datos.

Pandora FMS es un sistema de monitorización escalable que tiene multitud de funcionalidades para extender el alcance y volumen de información recogida prácticamente sin limites.

Monitorización basada en agentes software vs. monitorización remota

Podríamos dividir la monitorización en dos grandes grupos basándonos en la forma de recoger la información: monitorización basada en agentes software y monitorización remota.

La **monitorización basada en agentes** consiste en la instalación de un pequeño software que permanece corriendo en el sistema y obteniendo información **de forma local** mediante la ejecución de comandos y scripts.

La **monitorización remota** consiste en el uso de la red para ejecutar chequeos remotos hacia los sistemas, sin necesidad de instalar ningún componente adicional en los equipos que se quieren monitorizar.

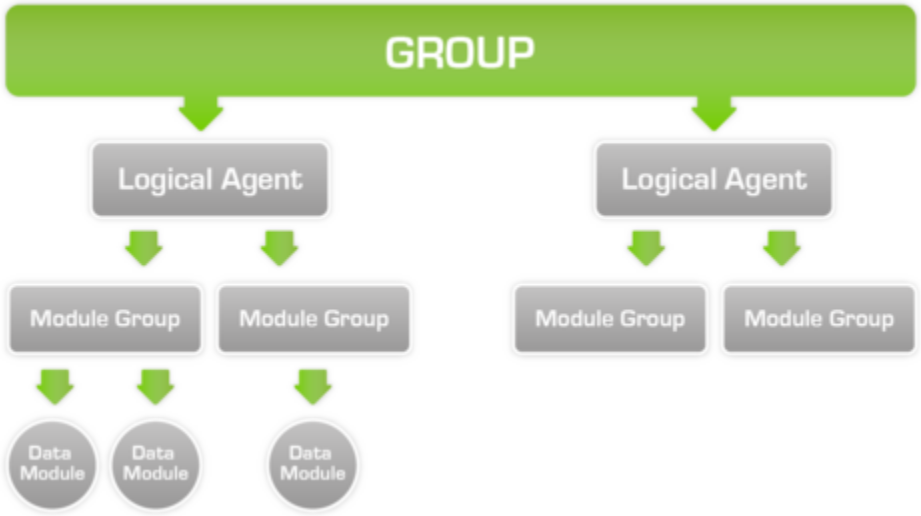
Como puede apreciarse, la monitorización basada en agentes obtendrá la información mediante **chequeos locales** mientras que la monitorización remota obtendrá la información mediante **chequeos de red** desde el servidor de Pandora FMS.

Con Pandora la monitorización puede ser de una u otra manera y también combinada, produciendo una monitorización mixta.

Agentes en Pandora FMS

La monitorización realizada por Pandora FMS se clasifica en *agentes*. Un agente siempre pertenece a un *grupo*. Estos agentes van a equivaler a cada uno de los distintos equipos, dispositivos, webs o aplicaciones que estemos monitorizando.

Los agentes definidos en la consola de Pandora FMS pueden presentar información local recogida mediante un agente software, información remota recogida mediante chequeos de red, o ambas cosas. Por ello, cabe destacar la diferencia entre agentes como unidad organizativa en la consola de Pandora FMS, y agentes software como servicios de recolección de datos locales.



Monitorización de estados

Cuando monitorizamos, obtenemos valores de un sistema, sea la memoria, CPU, temperatura del chasis, número de usuarios conectados, de pedidos en una WEB de comercio electrónico o cualquier otro valor numérico. A veces sólo nos interesa el dato, pero generalmente querremos asociar un ESTADO a esos valores, de forma que al superar un "UMBRAL" cambie de estado, para saber si algo está bien o está mal. Por eso cuando hablamos de monitorización, tenemos que introducir el concepto de ESTADO.

Pandora FMS permite definir **umbrales** para definir el estado que un chequeo tendrá basándose en los datos que muestre. Los tres estados posibles son: NORMAL, WARNING y CRITICAL. Un umbral es un valor a partir del cual pasamos de un estado a otro. El estado que adquirirán los módulos dependerá de estos umbrales, que se especifican mediante los siguientes parámetros presentes en la configuración de cada módulo:

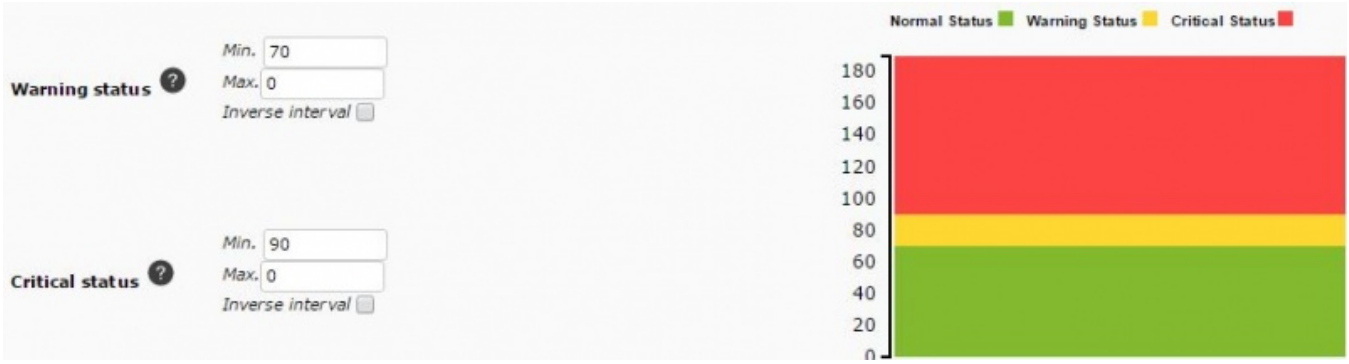
*__Warning status - Min. Max.__: límites inferior y superior para el estado warning. Si el valor numérico del módulo se encuentra entre este rango, el módulo pasará a estado warning. Si no se especifica límite superior éste será infinito (todo valor superior al límite inferior).
*__Warning status - Str.__: expresión regular para módulos tipo alfanumérico (string). Si se encuentran coincidencias el módulo pasará a estado warning.
*__Critical status - Min. Max.__: límites inferior y superior para el estado crítico. Si el valor numérico del módulo se encuentra entre este rango, el módulo pasará a estado critical. Si no se especifica límite superior éste será infinito (todo valor superior al límite inferior).
*__Critical status - Str.__: expresión regular para módulos tipo alfanumérico (string). Si se encuentran coincidencias el módulo pasará a estado crítico.
*__Inverse interval__: presente tanto para el umbral warning como critical. Si se encuentra activado, el módulo cambiará de estado cuando sus valores estén **f uera del intervalo** especificado en los umbrales. También funciona para módulos alfanuméricos (string), si las cadenas de texto NO coinciden con lo especificado en Warning/Critical Str., el módulo cambiará de estado.

En caso de los umbrales *warning* y *critical* coincidan en algún rango, siempre prevalecerá el umbral *critical*.

Umbrales numéricos - Caso práctico 1

Tenemos un módulo de porcentaje de uso de CPU que siempre va a estar verde en el estado de los agentes, ya que simplemente informa de un valor entre 0% y 100%. Si queremos que el módulo de uso de CPU pase a estado warning (color amarillo) cuando llegue al 70% de su uso, y a estado crítico (rojo) cuando llegue al 90%, deberemos configurar los umbrales de la siguiente forma

- Warning status Min.: 70
- Critical status Min.: 90



Con ello, cuando se llegue al valor 90 el módulo aparecerá en rojo (CRITICAL), mientras que entre 70 y 89.99 estará en amarillo (WARNING), y por debajo de 70 en verde (NORMAL).

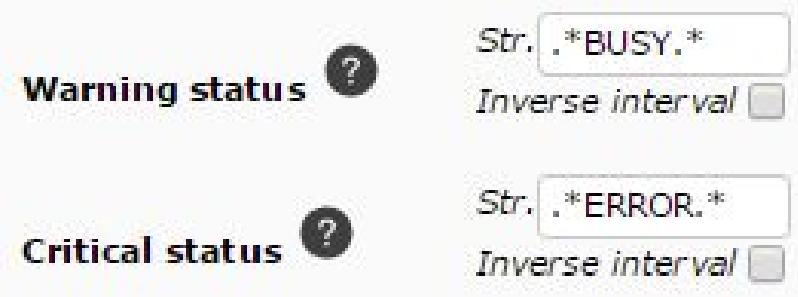
Debido al funcionamiento de los umbrales, en casos como este no es necesario establecer los límites superiores. Esto es debido a que si únicamente se establece el umbral inferior, el superior se tendrá en cuenta como "sin límite", por lo que todo valor por encima del umbral inferior será tenido en cuenta como dentro del umbral. Además, si los umbrales se cruzan, prevalecerá el umbral crítico sobre el warning, dando como resultado la gráfica de umbrales que mostrábamos en la captura anterior.

Umbrales de texto - Caso práctico 2

Si tenemos un módulo de tipo *string* podemos configurar los estados usando expresiones regulares en los campos *Str* de los parámetros *Warning Status* y *Critical Status*. En este caso tenemos un módulo que nos puede devolver como dato: *OK*, *ERROR connection fail* o bien *BUSY too many devices*, dependiendo del resultado de la consulta.

Para configurar los estados de WARNING y CRITICAL del módulo de texto usaremos las siguientes expresiones regulares:

Warning Status: .**BUSY.**
Cririca Status: .**ERROR.**



Con esta configuración el módulo tendrá estado WARNING cuando el dato contenga el string BUSY y su estado será CRITICAL cuando el dato contenga el string ERROR. **Debe tener cuidado porque las expresiones regulares son case sensitive**.

Monitorización dinámica (Umbrales automáticos)

La monitorización dinámica consiste en el ajuste dinámico y de forma automática de los umbrales de estados de los módulos de una forma inteligente y predictiva. El modo de funcionar es recoger los valores de un periodo determinado y calcular una media y una desviación estándar, que son utilizadas para establecer los umbrales correspondientes.

La configuración se realiza a nivel de módulo, y los parámetros posibles son:

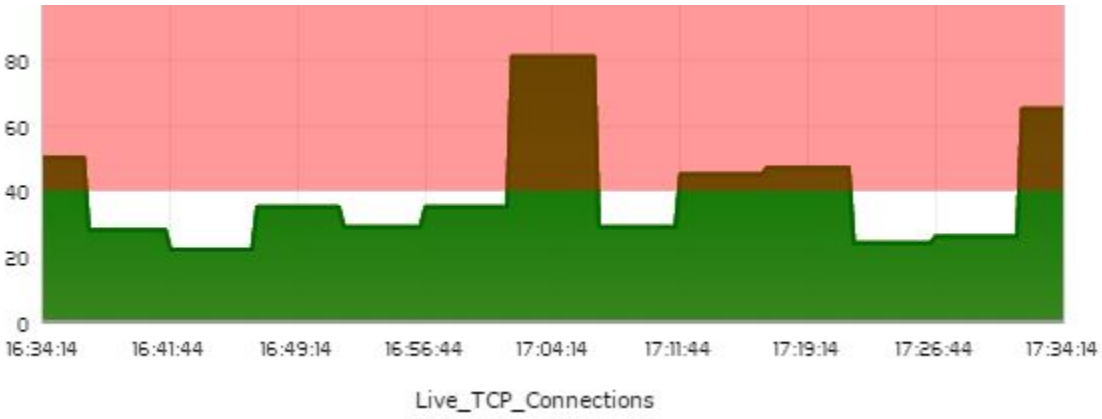
*__Dynamic Threshold Interval__: intervalo de tiempo que se considerará para realizar el cálculo de umbrales. Si elegimos 1 mes, el sistema tomará todos los datos existentes en el último mes y construirá los umbrales en base a esos datos.
*__Dynamic Threshold Two Tailed__: si se activa, el sistema de umbrales dinámicos establecerá también umbrales **por debajo** de la media. Si se encuentra desmarcado (por defecto) solo se establecerán umbrales con valores **por encima** de la media.
*__Dynamic Threshold Max.__: permite aumentar el límite superior en el porcentaje que indiquemos. Ej: si los valores promedio se encuentran alrededor del 60 y el umbral crítico ha sido establecido a partir del valor 80, si establecemos el valor *Dynamic Threshold Max: 10*, aumentaremos un 10% este umbral crítico, por lo que quedaría en un 88.
*__Dynamic Threshold Min.__: solo aplica si tenemos activo el parámetro *Dynamic Threshold Two Tailed*. Permite reducir el límite inferior en el porcentaje que indiquemos. Ej: si los valores promedio se encuentran alrededor del 60 y el umbral crítico inferior ha sido establecido en un valor 40, si establecemos el valor *Dynamic Threshold Min: 10*, reduciremos un 10% este umbral crítico, por lo que quedaría en un 36.

Además existen varios parámetros de configuración adicionales en el fichero *pandora_server.conf*.

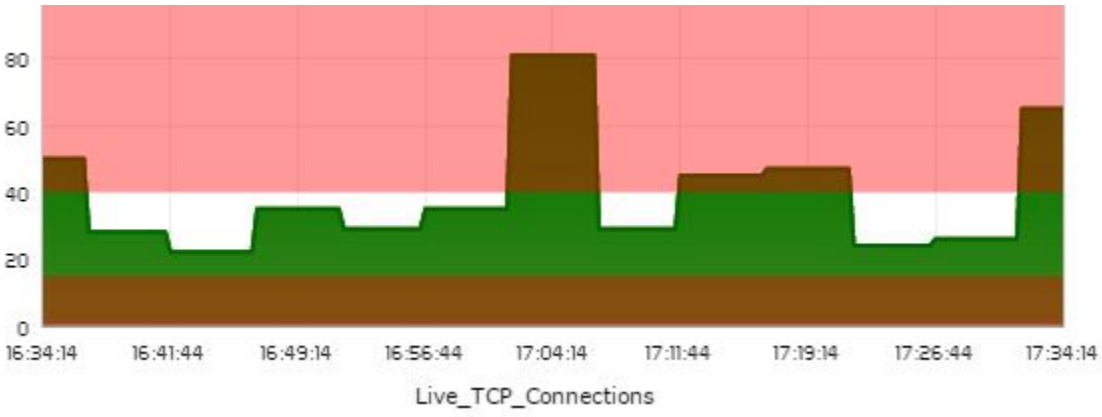
*__dynamic_updates__: este parámetro determina cuántas veces se recalculan los umbrales durante el periodo de tiempo establecido en *Dynamic Threshold Interval*. Si configurásemos *Dynamic Threshold Interval* con un valor de 1 semana, por defecto se hace recogen los datos de una semana hacia atrás y se hace el cálculo una única vez, repitiéndose el proceso nuevamente una vez transcurrida una semana. Si modificásemos el parámetro *dynamic_updates* podríamos incrementar esta frecuencia. Por ejemplo configurar el parámetro con un valor de 3 hará que se recalculen los umbrales hasta tres veces durante el periodo de una semana (o el periodo que tengamos configurado en *Dynamic Threshold Interval*. Su valor por defecto es 5.
*__dynamic_warning__: diferencia entre los umbrales warning y critical, en porcentaje. Su valor por defecto es 25.
*__dynamic_constant__: determina la desviación de la media que se utilizará para establecer los umbrales, valores mayores harán que los umbrales estén más alejados de los valores promedios. Su valor por defecto es 10.

En el siguiente ejemplo el valor promedio calculado se encuentra a la altura de la línea roja (aprox. 30):

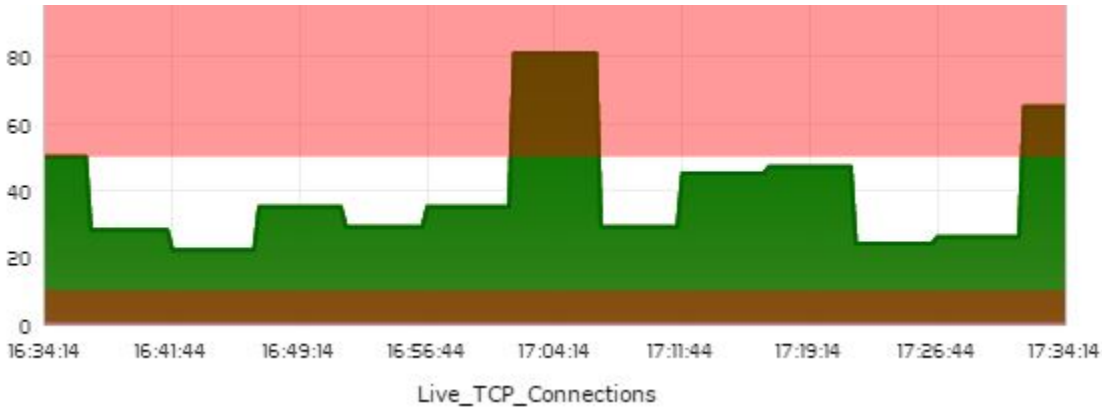
Al activar los umbrales dinámicos, se ha establecido de este modo el umbral superior (aprox. 45 y superiores):



Hemos activado el parámetro *Dynamic Threshold Two Tailed*, de modo que también se ha establecido un umbral crítico por debajo de los valores promedios (aprox. 15 e inferiores):

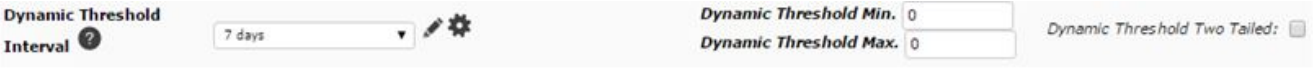


Ahora hemos establecido los parámetros *Dynamic Threshold Min.* y *Dynamic Threshold Max.* a 20 y 30 respectivamente, por lo que los umbrales se han abierto, siendo ligeramente más permisivos:



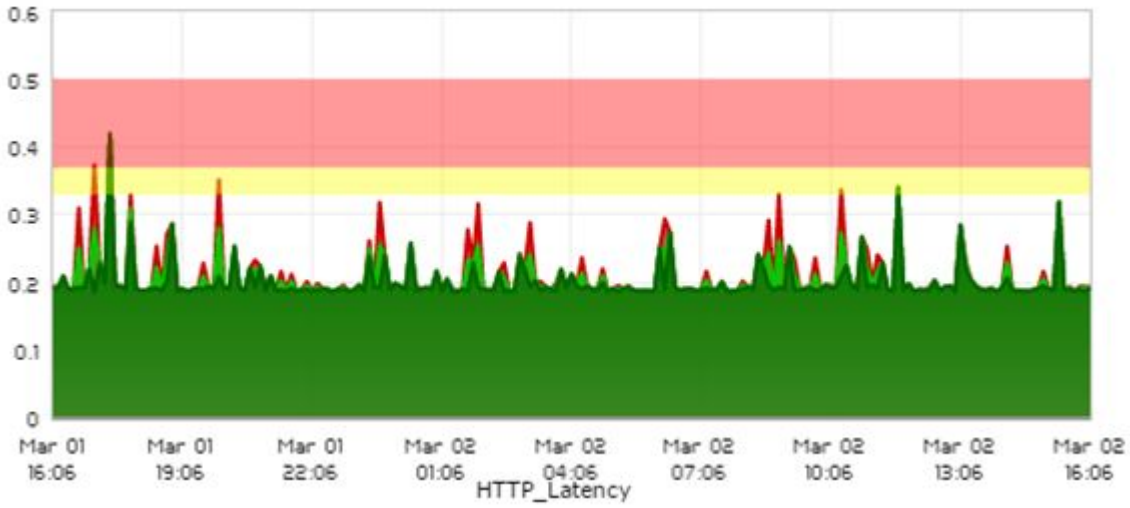
Caso práctico 1

Partimos de un módulo de latencia web. La configuración básica que hemos empleado toma en cuenta un intervalo de una semana:



Al guardar cambios, tras ejecutarse *pandora_db* los umbrales se han establecido de este modo:

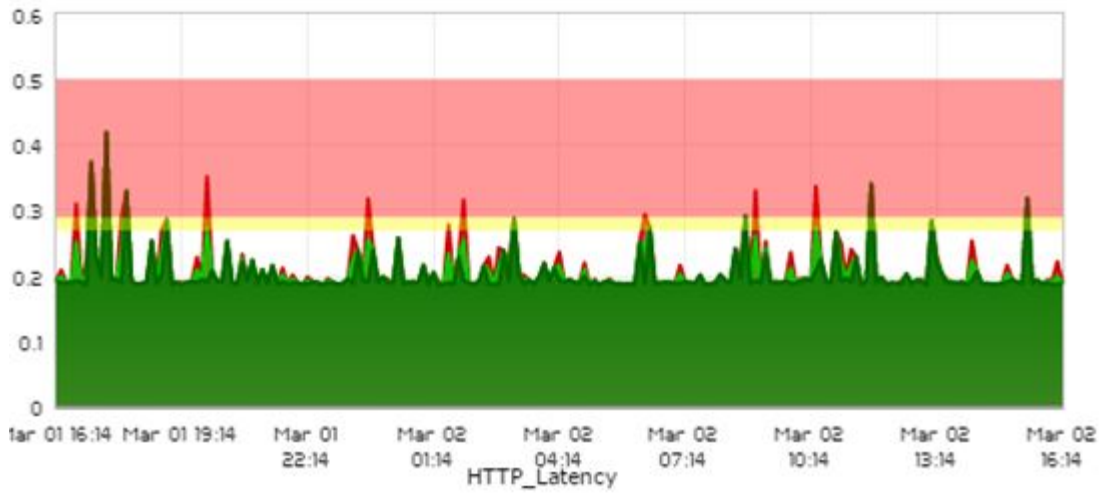
El módulo por tanto cambiará a estado *warning* cuando la altencia sea superior a 0'33 segundos, y a *critical* cuando sea superior a 0'37 segundos. En la gráfica se muestra del siguiente modo:



Se ha considerado que el umbral es algo permisivo, por lo que se decide hacer uso del parámetro *Dynamic Threshold Min.* para reducir los umbrales mínimos. Como en este caso el umbral no tiene valores máximos porque se considerará incorrecto todo lo que sea superior a un determinado valor, no vamos a utilizar *Dynamic Threshold Max.*. La modificación hecha es así:

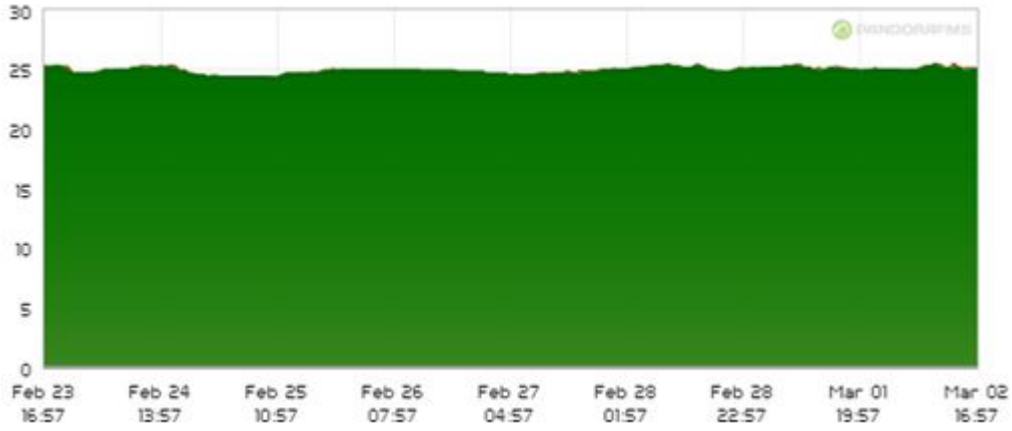
Tras aplicar cambios y ejecutarse el *pandora_db*, los umbrales quedan establecidos del siguiente modo:

Y la gráfica tendrá este aspecto:



Caso práctico 2

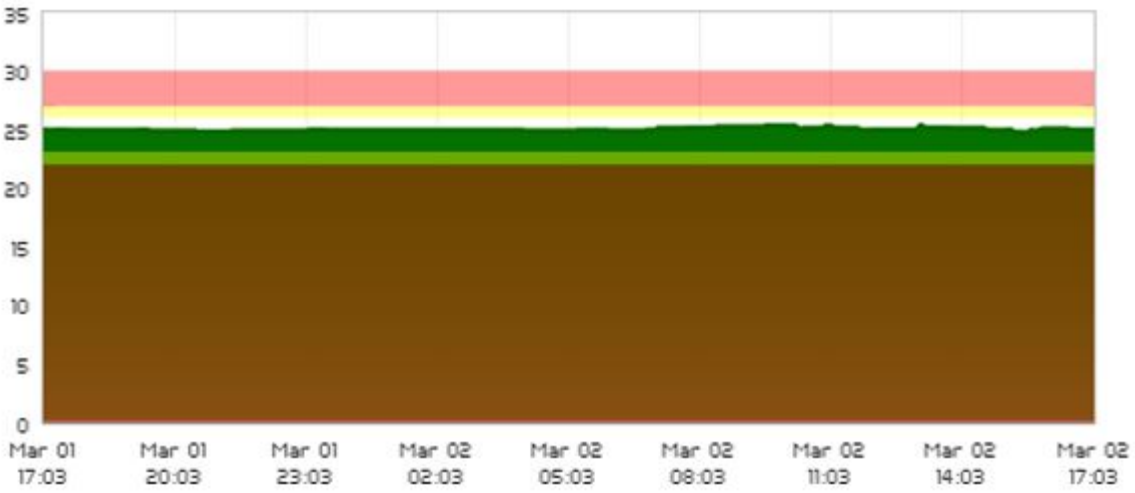
En este ejemplo estamos monitorizando la temperatura de una sala de control o un CPD, la gráfica que muestra presenta unos valores con poca variación:



En esta situación es fundamental que la temperatura se mantenga estable y no alcance valores muy superiores pero tampoco muy inferiores, por lo que utilizaremos el parámetro *Dynamic Threshold Two Tailed* para delimitar umbrales tanto por encima como por debajo. La configuración es la siguiente:

Los umbrales que se han generado automáticamente han sido estos:

La gráfica los muestra de este modo:



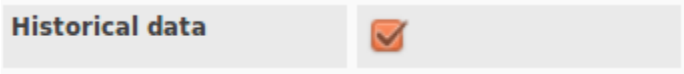
De este modo se considerará normal todos los valores que se encuentren entre 23'10 y 26, ya que es la temperatura aceptable en nuestro CPD o sala de control. De necesitarlo, podríamos de nuevo utilizar los parámetros *Dynamic Threshold Min.* y *Dynamic Threshold Max.* para ajustar los umbrales si fuese necesario.

Otros conceptos de monitorización básicos

Intervalo de monitorización

Todos los sistemas *síncronos* de monitorización necesitan definir "cada cuanto" se realiza una prueba. Pueden ser 5 segundos, 5 minutos o cinco horas. Pero hay que definirlo. A frecuencias de tiempo más cortas, más volumen de información se almacena. Un dato que se procesa cada 5 minutos, tendrá al cabo del día, 5x12x24=288 muestras, mientras que un dato que se procesa cada 30 segundos, tendrá 2880 muestras. De cara a entender cuanta información va a manejar nuestro sistema de monitorización es clave entender este concepto que en Pandora FMS denominamos simplemente *intervalo*

Histórico

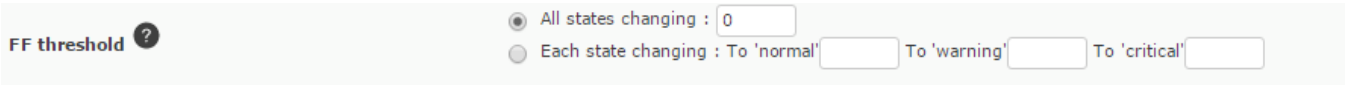


Pandora FMS permite almacenar (opcionalmente) el histórico de cada dato, de forma individual. Por defecto todos los módulos guardan histórico (lo que les permite pintar gráficas, incluirlos en informes de tipo histórico/evolutivo, etc). Sin embargo en una implantación muy grande que necesite monitorizar muchos datos, puede que pueda prescindir de mantener el histórico de algunos de los datos, permitiendo así utilizar menos recursos.

Esta opción permite desactivar el histórico de aquellos módulos donde no necesite guardar un histórico. Aunque desactive el histórico, las alertas continuarán funcionando exactamente igual, lo mismo que la generación de eventos y la visualización del estado actual de ese monitor.

Flip-Flop

Se conoce por FlipFlop (FF) a un fenómeno usual en monitorización: cuando un valor oscila de forma frecuente entre valores alternativas (MAL/BIEN) que dificulta su interpretación. Cuando esto ocurre, se suele emplear un "umbral" de forma que para considerar que algo ha cambiado de estado, tiene que "permanecer" más de X intervalos seguidos en un estado sin alterarse. A esto lo llamamos en terminología de Pandora FMS: *FF Threshold*



El parámetro FF Threshold (FF = FlipFlop) se utiliza para "filtrar" los cambios continuos de estado en la generación de eventos/estados, de forma que se le pueda decir a Pandora FMS que hasta que un elemento no esté al menos X veces en el mismo estado después de cambiar desde un estado original, no lo considere como que ha cambiado. Pongamos un ejemplo clásico: Un ping a un host donde hay pérdida de paquetes. En un entorno de este tipo, podría darnos resultados como:

1
1
0
1
1
0
1
1
1

Sin embargo el host está vivo en todos los casos. Lo que queremos realmente decirle a Pandora es que hasta que el host no diga que está al menos tres veces caído, no lo marque como tal, de forma que en el caso anterior nunca estaría como caído, y solo en este caso lo estaría:

1
1
0
1
0
0
0

A partir de este punto ya lo marcaría como caído, pero no antes.

Por tanto la protección anti Flip-Flop sirve para evitar estas molestas fluctuaciones. Todos los módulos lo implementan y su uso es para evitar el cambio de estado (delimitado por sus limites definidos o limites automáticos, como es el caso de los módulos *proc).

Volver al Indice