Diesen Beitrag schrieb ich vor 3 Jahren. Behalte das beim Lesen bitte im Hinterkopf.

Mit dem gestrigen Git-Checkout landete ein `icinga2-module-director` auf meinem System, mit dem das Erstellen und Deployen von *notifications* erstmals komplett funktionierte und eine syntaktisch korrekte Konfiguration ergab. Benachrichtigungen flexibel übers Webinterface zusammenklicken? Es funktioniert — ich zeige euch, wie!

Update 19. Juni 2017: Die Scripte dieses Artikels entstanden aus der Not heraus, dass der Icinga Director nicht mit `ENV`-Variablen umgehen kann. Die Resonanz darauf hat mich mehr als überrascht: wahnsinnig viele Leute zeigten sich wahnsinnig interessiert an meiner Art, Notifications zu erstellen. Nachdem ich nun also den Umgang mit `GIT` ansatzweise kapiert habe, habe ich den ersten Pull Request meines Lebens gestellt, und ab der bald verfügbaren 2.7.0 von Icinga 2 werden meine Scripte der neue Default sein. Deshalb friere ich den vorliegenden Artikel an dieser Stelle ein und bitte euch, Fragen und Diskussionen über die offiziellen Kanäle zu führen :-)

Time period

Klar — im ersten Schritt sorgst du natürlich für ein ganz frisches `git clone` des Moduls. Beginnen wir nun also damit, dass wir uns (in Abhängigkeit vom Template *generic-director-timeperiod*) ein erstes *time period*-Objekt erstellen — ich nenne meins einfach *Immerzu* und setze als *time range* für jeden Tag der Woche die vollen 24 Stunden ein.

User

Benachrichtigungen brauchen natürlich auch einen User, der sie empfängt — abhängig von meinem Template *generic-director-user* (das selbstverständlich `enable_notifications = true` setzt!) erstelle ich nun also mich selbst — das ist selbsterklärend.
Im nächsten Schritt wird es nun schon spannender: es wird nämlich ein Script benötigt, das den Emailversand übernimmt…

Die Scripte

D die allseits bekannten `mail-host-notification.sh` und `mail-service-notification.sh` in Kombination mit ihren *NotificationCommand*-Definitionen sind so nicht in den Director übertragbar — dort gibt es ~~(noch?)~~ keinen `ENV`-Support. Dass dieser auch nicht zwingend benötigt wird, zeige ich euch anhand der folgenden Scripte, die ich mir gehäkelt habe — sie gewinnen ganz sicher keinen Schönheitswettbewerb, aber ihr dürft sie gerne nutzen.

```
$ cd /etc/icinga2/scripts
$ wget
https://raw.githubusercontent.com/sysadmama/misc/master/icinga2/scripts/host-by-mail.sh
```

```
$ wget
https://raw.githubusercontent.com/sysadmama/misc/master/icinga2/scripts/servi
ce-by-mail.sh
$ chmod +x *by-mail.sh
```

Notification Plugin Command

Anhand der nun vorhandenen Scripte können zwei neue Kommando-Objekte vom Typ *Notification Plugin Command* erstellt werden:

```
object NotificationCommand "Service Alarm By Email" {
    import "plugin-notification-command"
    command = [ "/etc/icinga2/scripts/service-by-mail.sh" ]
arguments += {
        "-4" = {
            required = true
            value = "$address$"
        }
        "-6" = "$address6$"
        "-b" = "$notification.author$"
        "-c" = "$notification.comment$"
        "-d" = {
            required = true
            value = "$icinga.long_date_time$"
        }
        "-e" = {
            required = true
            value = "$service.name$"
        }
        "-f" = "$notification_from$"
        "-i" = "$icingaweb2url$"
        "-l" = {
            required = true
            value = "$host.name$"
        }
        "-n" = {
            required = true
            value = "$host.display_name$"
        }
        "-o" = {
            required = true
            value = "$service.output$"
        }
        "-r" = {
            required = true
            value = "$user.email$"
        }
        "-s" = {
            required = true
            value = "$service.state$"
        }
```

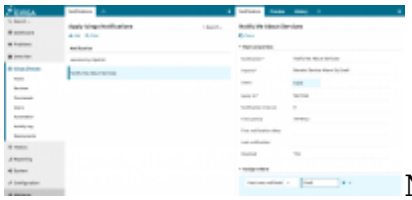
```
        "-t" = "$notification.type$"
        "-u" = {
            required = true
            value = "$service.display_name$"
        }
        "-v" = "$notification_logtosyslog$"
    }
}

object NotificationCommand "Host Alarm By Email" {
    import "plugin-notification-command"
    command = [ "/etc/icinga2/scripts/host-by-mail.sh" ]
    arguments += {
        "-4" = {
            required = true
            value = "$address$"
        }
        "-6" = "$address6$"
        "-b" = "$notification.author$"
        "-c" = "$notification.comment$"
        "-d" = {
            required = true
            value = "$icinga.long_date_time$"
        }
        "-f" = "$notification_from$"
        "-i" = "$icingaweb2url$"
        "-l" = {
            required = true
            value = "$host.name$"
        }
        "-n" = {
            required = true
            value = "$host.display_name$"
        }
        "-o" = {
            required = true
            value = "$host.output$"
        }
        "-r" = {
            required = true
            value = "$user.email$"
        }
        "-s" = {
            required = true
            value = "$host.state$"
        }
        "-t" = {
            required = true
            value = "$notification.type$"
        }
        "-v" = "$notification_logtosyslog$"
```

```
    }
}
```

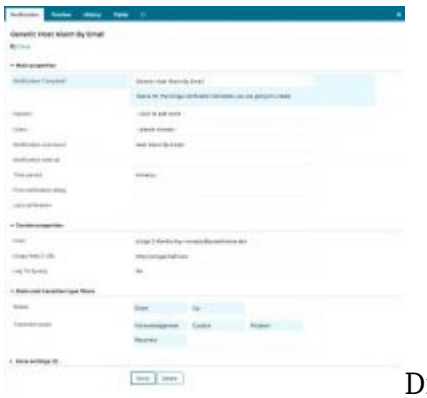
Sowohl das *command* `Host Alarm By Email` als auch `Service Alarm By Email` werden nun innerhalb des Director angelegt: Objekte vom Typ *Notification Plugin Command*, die auf das jeweils korrespondierende Script im Filesystem verweisen. Als Argumente werden die im Script definierten Parameter übernommen, und als Wert erhalten diese den jeweils gewünschten Wert über die *icinga runtime macros*, der zur Laufzeit ermittelt wird. So bezeichnet `-4` beispielsweise die IPv4-Adresse des Hosts, und diese wird über das *runtime macro* `$address$` übergeben. Häkelt euch eure ganz individuelle Benachrichtigungs-Lösung — ihr werdet schnell merken, wie umfangreich eure Möglichkeiten sind!

(Einschub: beim Erfassen der Parameter erhalte ich derzeit immer die rot hinterlegte Meldung »Trying to get invalid property „argument_name“« — was der Funktionalität jedoch keinen Abbruch zu tun scheint. Muss ich mal’n Bugreport aufmachen oder so. Einschub Ende.) → Der Bug ist inzwischen behoben.

Notification Object



Nach all der Vorarbeit sind wir nun an dem Punkt angelangt, an dem wir die eigentliche Benachrichtigung erfassen können; im ersten Schritt werden zwei Templates (`Generic Service Alarm By Email` und `Generic Host Alarm By Email`) erstellt — ich definiere mir hier *states* und *transition types*, wie ich sie haben möchte. Und nun kann ich eine *apply rule* erfassen — wichtig ist hier, dass der Bereich »Assign where« erst dann auftaucht, **nachdem** auf *Store* geklickt wurde! Über »Assign where« kann nun definiert werden, bei welchen Events der User informiert werden soll; die Auswahl »Host« oder »Service« unter »Apply to« ist hierbei zwingend erforderlich.



Die inzwischen massiv erweiterte Version des Scripts gibt nun auch die Möglichkeit, einen `From:`-Header zu definieren — legt man sich ein Datenfeld im Director an, ist auch das bequem über das Webinterface steuerbar. Wird für `$icingaweb2url` ein String hinterlegt, erweitert sich die Ausgabe der Email um den passenden Link. Das Script fängt nun ab, wenn keine Kommentare vorhanden sind, und blendet die Zeile nur noch im Bedarfsfall ein. Und per `Log To Syslog` lässt sich eine Debug-Ausgabe ins `syslog` realisieren, wenn man sie denn haben möchte.

Fazit

Mein Hirn brauchte eine Weile, um hier durchzusteigen; im Nachhinein finde ich es eigentlich sehr logisch und einfach ;-) Nun können die Testläufe beginnen — testet ausgiebig, denn gerade bei Benachrichtigungen möchte man keine bösen Überraschungen erleben. Noch spannender wird die Sache, wenn alternative Benachrichtigungen hinzukommen — Jabber, HipChat, Telegram, SMS — damit möchte ich auch noch ein wenig rumbasteln.

Aber für heute freue ich mich einfach, dass das Erfassen der User und ihrer Benachrichtigung nun über den Director funktioniert — und mir die Arbeit in Zukunft signifikant erleichtern wird.