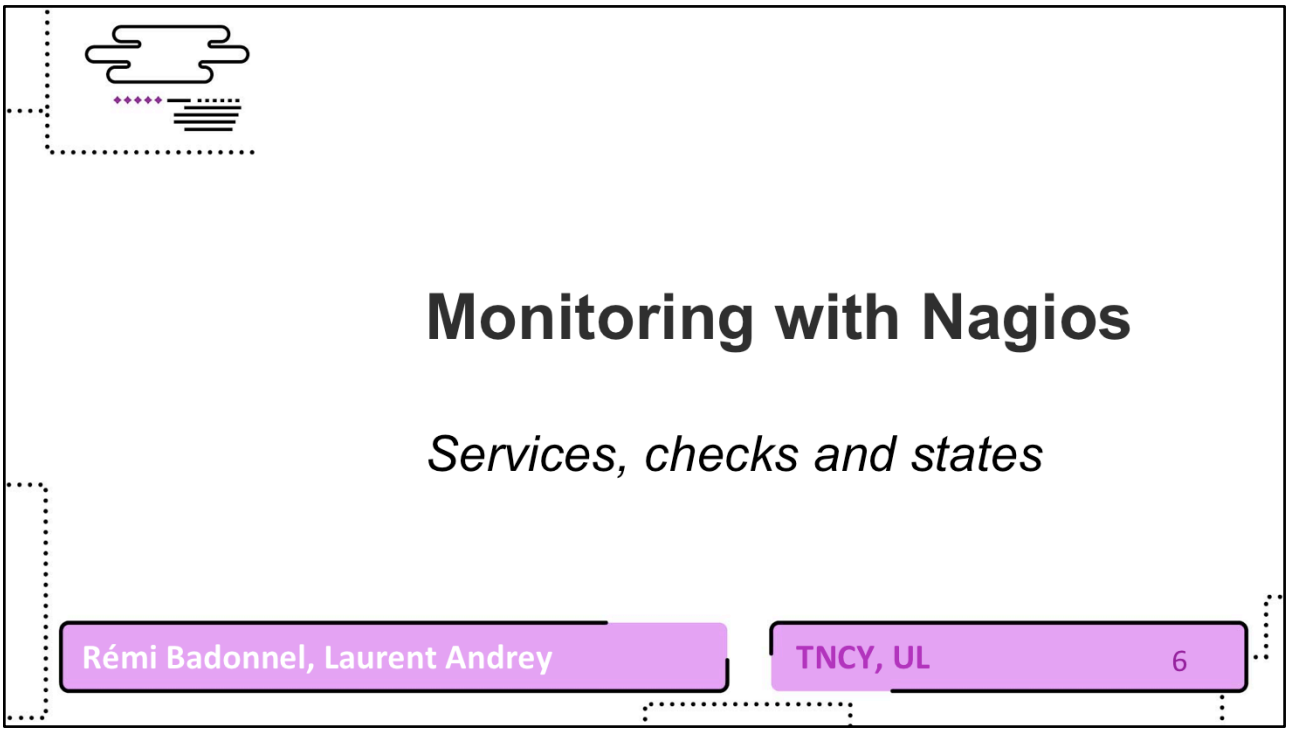
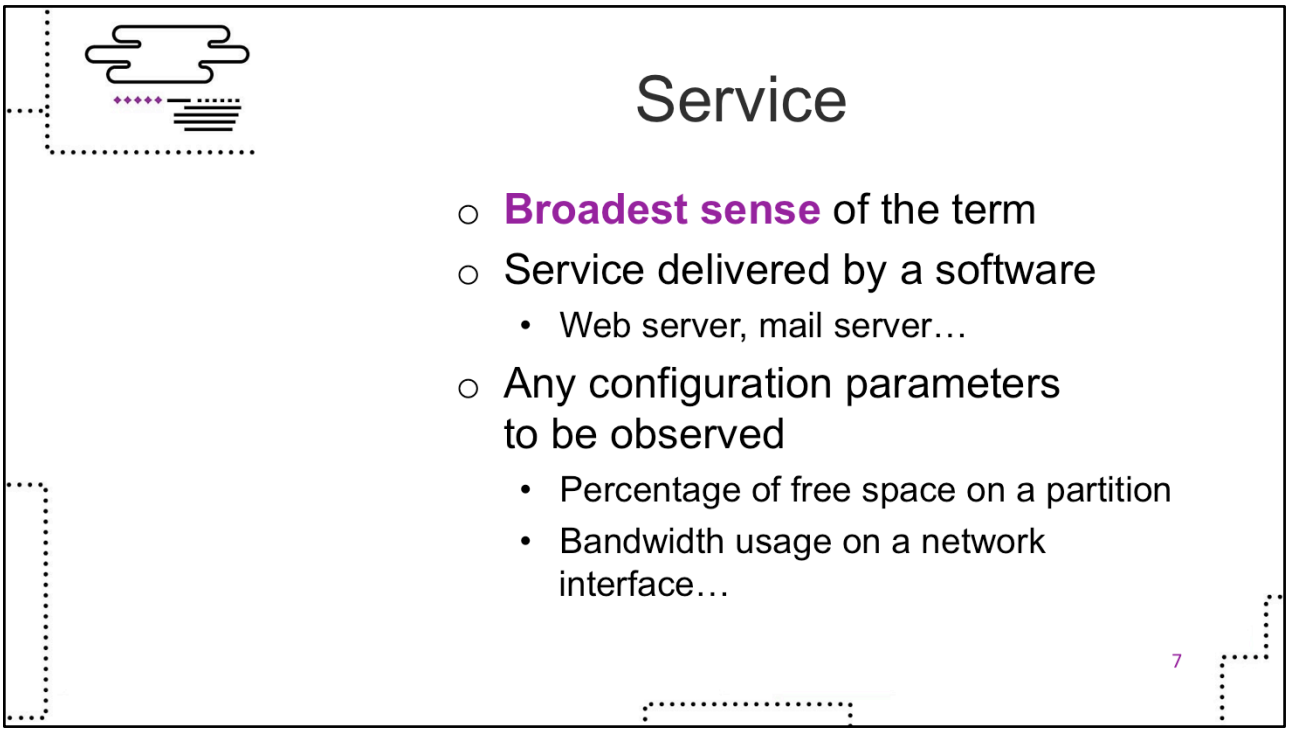
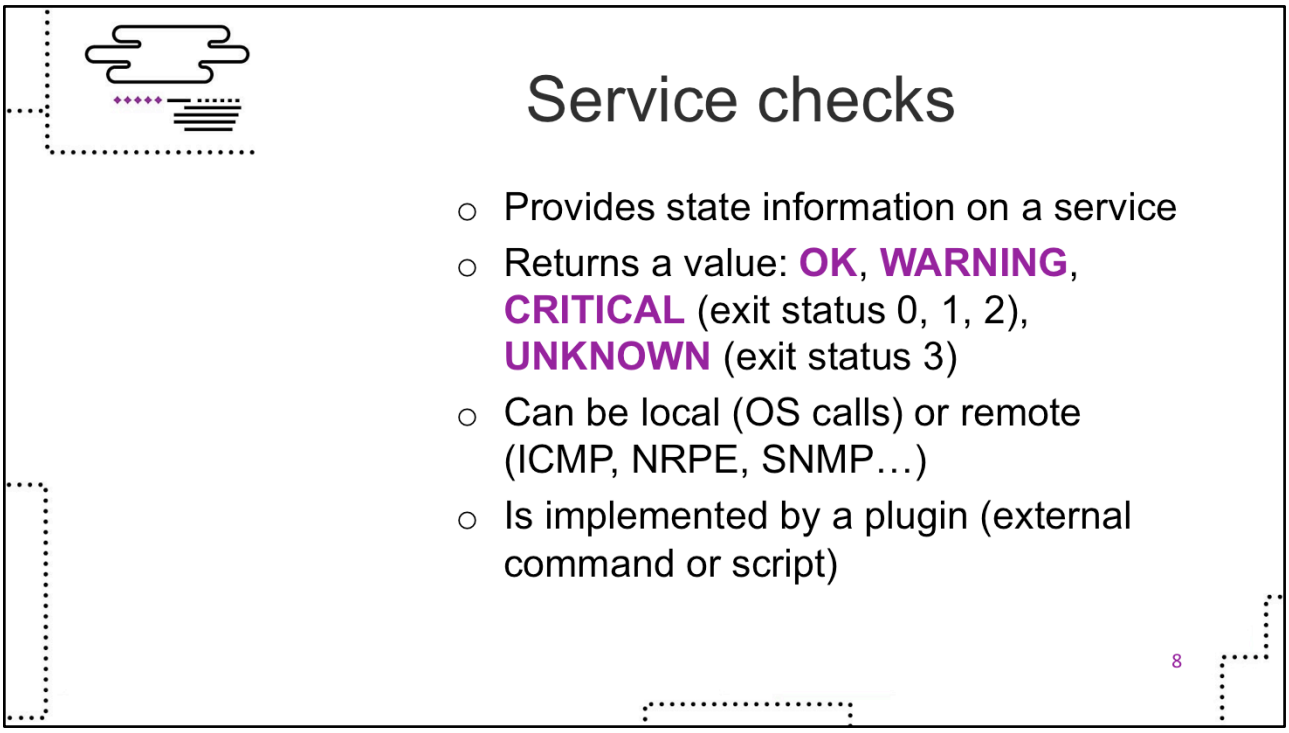


Cette seconde leçon sur Nagios va permettre d'approfondir les notions de services, de tests et d'états de services, et de comprendre comment ces trois concepts sont liés.



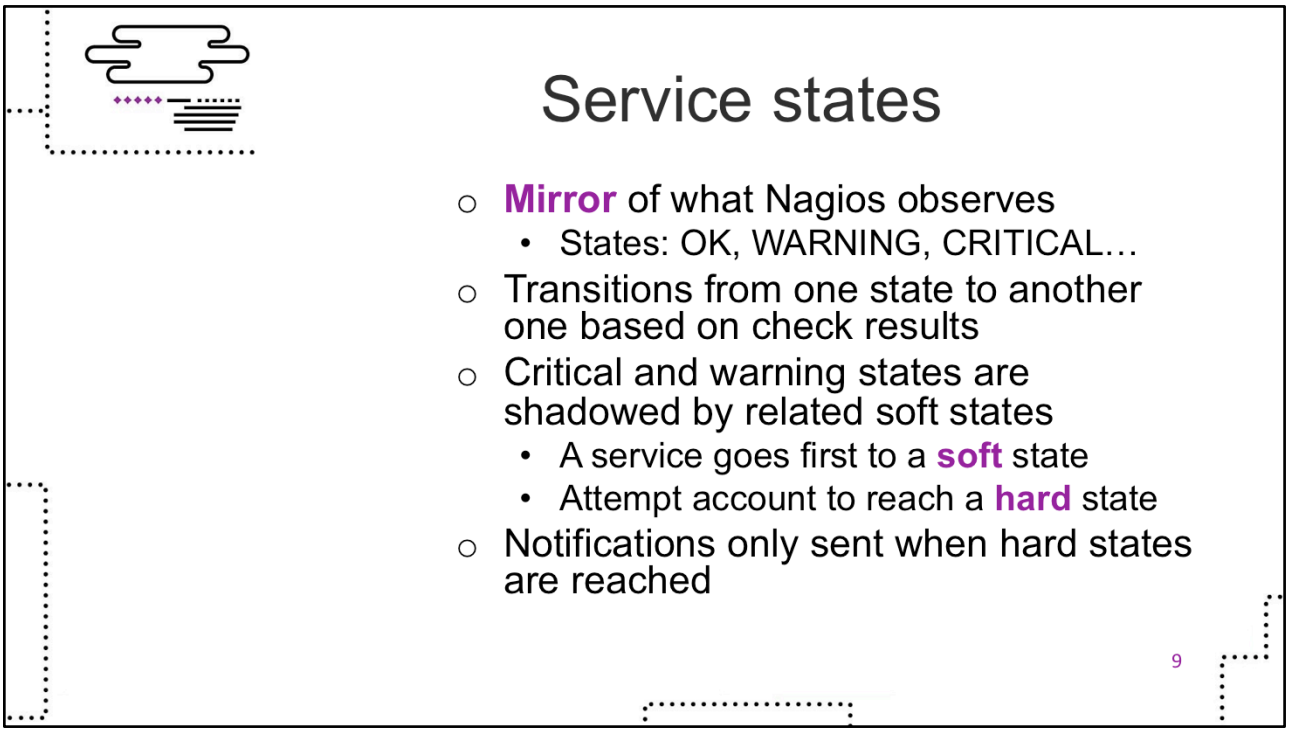
Pour utiliser Nagios, il est tout d'abord important de comprendre ce que l'on entend par service. Cette notion s'entend au sens large. Sous Nagios, une machine est associée à un ensemble de services. Un service peut être un service fourni par un logiciel, tel qu'un serveur web ou un serveur de messagerie. Il peut aussi correspondre à tout paramètre de configuration qui peut être observé sur un équipement, tel que le pourcentage d'espace disponible sur une partition, la quantité de bande passante utilisée sur une interface réseau, ou encore le nombre de processus fonctionnant sur une machine. Ce qui est important, c'est de pouvoir inférer, à partir de ces paramètres, l'état du service. Et ce rôle d'inférence est en fait assuré par les tests.



En effet, les tests retournent des informations sur l'état des différents services. Plus précisément, ils retournent un code de sortie, qui va les caractériser. Par exemple, un code de sortie (ou exit status) de 0 correspond à un état OK. Un exit status de 1 va indiquer un état WARNING. Et un exit status de 2 va indiquer un état CRITICAL. Il existe également un état UNKNOWN correspondant à l'exit status 3, qui va permettre d'indiquer un timeout (expiration de délai) ou une erreur au cours de l'exécution du plugin.

Ces tests peuvent être exécutés localement au travers d'appels système, ou de façon distante en s'appuyant sur les protocoles réseaux (comme ICMP), sur des agents dédiés (je parlais tout à l'heure des agents Nagios NRPE). Il est également possible d'intégrer des agents SNMP en utilisant des plugins spécifiques. Les tests sont implantés par des plugins, correspondant à des programmes ou des scripts.

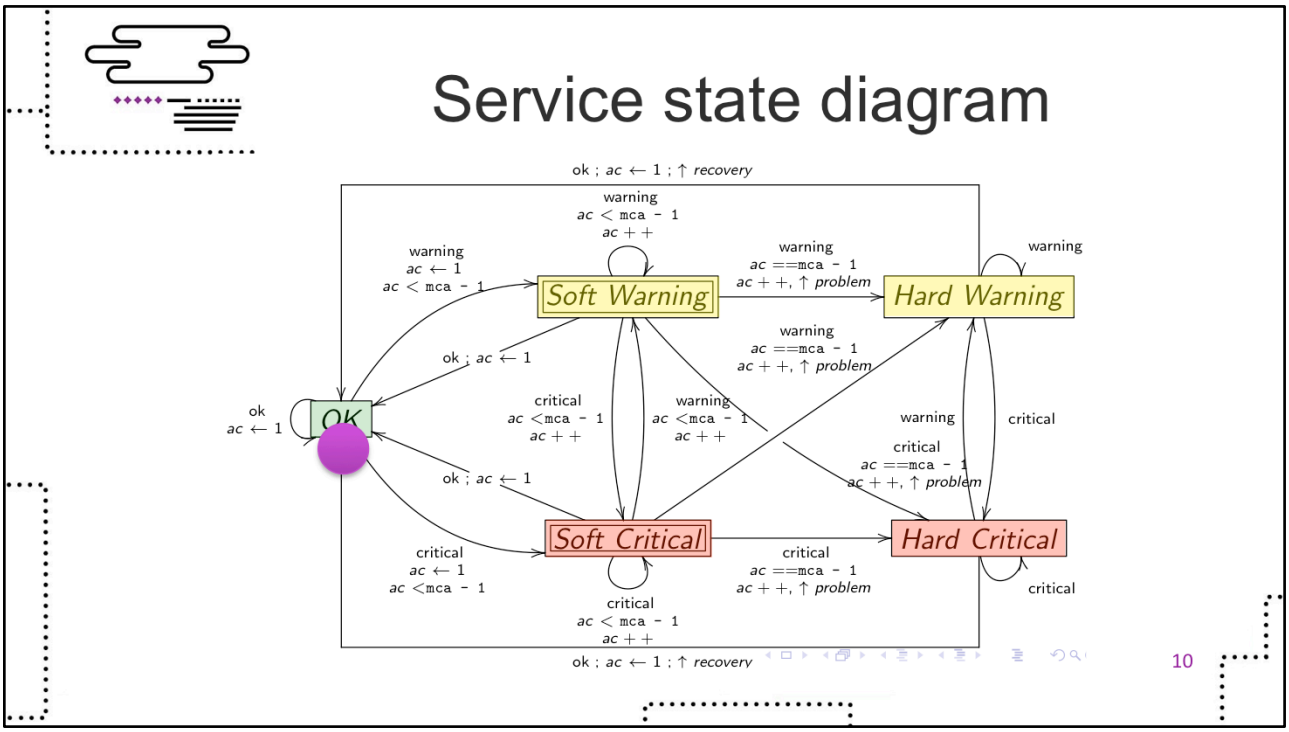
Il est facile de créer ses propres plugins, en développant un script qui va prendre en entrée des paramètres de configuration incluant des valeurs seuils, et qui va retourner le bon code de sortie pour indiquer si le service fonctionne correctement ou non.



L'état d'un service correspond au miroir de ce que Nagios observe sur une infrastructure. Comme nous l'avons déjà mentionné, il existe quatre états principaux pour un service, à savoir les états OK, WARNING, CRITICAL et UNKNOWN. Des transitions d'un état à un autre sont possibles et sont provoqués par les résultats retournés par les tests. Dans le cas des états CRITICAL et WARNING, les états vont être dédoublés.

En fait, vous allez avoir un dédoublement, en un état SOFT et un état HARD qu'on peut aussi qualifier d'état confirmé. Concrètement, cela signifie que plusieurs tests successifs doivent retourner un même état donné, avant que ce nouvel état soit considéré comme confirmé. Un service va donc d'abord passer dans un état SOFT, (qui va être SOFT WARNING ou SOFT CRITICAL), et ensuite atteindre un état confirmé (qui va être HARD WARNING ou HARD CRITICAL) seulement après un certain nombre de résultats WARNING ou respectivement CRITICAL.

Les notifications seront alors uniquement envoyées aux administrateurs, ou aux usagers, lorsque les états HARD seront atteints.



Nous présentons ici, sous la forme d'un diagramme simplifié, la manière avec laquelle Nagios calcule l'état d'un service en fonction des résultats des tests. Nous retrouvons sur ce diagramme un état OK, qui est présenté ici en vert et qui est unique. Il s'agit en fait d'un état HARD. On a également deux états WARNING, qui sont présentés ici en jaune, correspondant respectivement à un état SOFT et un état HARD, et 2 états CRITICAL en rouge, correspondant à nouveau à un état SOFT et un état HARD. Les transitions sont provoqués par les résultats des tests, qui sont dans notre cas OK, CRITICAL ou WARNING.

Le diagramme utilise deux variables importantes : ac et mca. La variable ac (pour attempt count) est utilisée pour mémoriser le nombre de fois qu'un test a été exécuté. La variable mca (signifiant maximum check attempts) détermine le nombre de fois qu'un test doit être exécuté avant de passer dans un état HARD. Enfin, les flèches verticales sur les transitions représentent l'envoi de notifications. Cela inclut les notifications concernant des problèmes : état CRITICAL ou WARNING, et également celles concernant le rétablissement d'un service (typiquement, lorsqu'un service repasse dans l'état OK).

Considérons maintenant un scénario simple où un service est dans un état OK. Si le test d'un service retourne une valeur CRITICAL, le diagramme passe alors dans un état SOFT CRITICAL et la valeur de la variable attempt count (ac) est incrémentée. L'état n'est pas encore confirmé. Plusieurs tests vont être requis, avant d'atteindre l'état HARD correspondant. Dans notre scénario, la variable maximum check attempts (mca) est fixée à 4. Cela signifie que trois tests additionnels, avec la même valeur de retour, à savoir CRITICAL, vont être requis avant d'atteindre l'état CRITICAL HARD. Lorsque cet état est atteint, une notification est alors envoyée à l'administrateur.

La fréquence des tests n'est pas la même, selon que l'on est dans un état SOFT ou un état HARD. Un état SOFT est un état temporaire, qui requiert des tests additionnels pour être confirmé en un état HARD. Si bien que la fréquence de ces tests est habituellement plus élevée que pour un état HARD. Par exemple, dans notre scénario, l'intervalle de temps entre deux tests pour l'état SOFT pourrait être d'une minute, tandis que l'intervalle pourrait être de cinq minutes, lorsque l'état HARD est atteint. Le dédoublement d'états en un état SOFT et un état HARD permet d'éviter les biais qui pourraient survenir en cas de micro-pannes ou de micro-coupures réseau. Lorsque le test retourne la valeur OK, Nagios considère qu'un seul test est nécessaire pour passer dans l'état OK (qui est considéré comme confirmé).