LINUX ALS SERVER

Beveiliging met permissies

Deze keer in de reeks 'Linux als server' een klassiek onderwerp: permissies. Permissies vormen de basis van een goede beveiliging en in dit artikel geef ik een overzicht wat allemaal moet gebeuren om je Linux server goed te beveiligen met behulp van permissies. **Sander van Vugt**

We besteden aandacht aan drie verschillende sets van permissies. Als eerste lees je hoe de read, write en execute permissies gebruikt worden om de basisbeveiliging te regelen. Daarnaast besteden we ook kort aandacht aan de speciale permissies SUID, SGID en Sticky bit en aan access control lists.

USER EN KERNEL SPACE EN PERMISSIES

Voordat we aandacht besteden aan permissies zelf, eerst een kort woordje over user en kernel space. Zo kan je begrijpen waarom het voor de gebruiker root niet uitmaakt welke permissies op bestanden zijn toegepast. Kernel space is het onderdeel van het Linux besturingssysteem waar de kernel woont. In kernel space is onbeperkt toegang tot alle hardware op een systeem en tellen normale beveiligingsoplossingen, zoals Linux permissies, niet. Root is de kernel space user, en om die reden kan root overal bij, zelfs als root geen enkele permissie heeft.

User space is waar normale processen en gebruikers opereren. Om de werking hiervan goed te laten plaatsvinden, wordt gebruik gemaakt van Linux permissies. Deze permissies worden uitgedeeld in drie sets: de basis permissies, de geavanceerde permissies en de access control lists

BASIS PERMISSIES

Om te bepalen wie wat mag met een bestand of directory, wordt gebruik gemaakt van user, group en others. De user is de gebruiker die de eigenaar is. De groep is de groep die eigenaar is van een bestand en others verwijst naar alle andere gebruikers. Permissies worden toegekend aan deze user, group en others. Daarbij is het goed te weten dat slechts één user en één group aan een bestand toegekend kunnen worden.

De basispermissies zijn read, write en execute. Indien read is toegekend aan een bestand, dan betekent dit dat het bestand gelezen kan worden. Read op een directory zorgt ervoor dat de inhoud van de directory getoond wordt (zonder dat dit ook betekent dat bestanden in die directory gelezen kunnen worden). Als de write permissie op een bestand is toegekend, dan kan het bestand gewijzigd worden door de entiteit (user, group of others dus) die die permissie heeft. Indien toegepast op een directory, kunnen bestanden in die directory aangemaakt of verwijderd worden.

De basispermissies zijn read, write en execute

De execute permissie speelt een speciale rol. Deze permissie is nodig bij bestanden die uitvoerbare code bevatten om te zorgen dat het bestand ook inderdaad uitgevoerd mag worden. Execute op een directory zorgt ervoor dat een gebruiker met behulp van het cd commando (of een grafische file navigator) de directory kan activeren. Execute op een directory kan daarom als essentieel beschouwd worden. En om die reden zie je deze permissie altijd op directories waarop de read permissie is toegepast.

BEHEER VAN PERMISSIES

Om permissies te beheren, moet je eerst het eigenaarschap instellen. Dit gebeurt met behulp van het chown of chgrp commando. Met chown stel je de user en/of groepeigenaar van een bestand in, waarbij chgrp alleen gebruikt wordt om de groepeigenaar in te stellen. Gebruik bijvoorbeeld **chown linda mijnbestand** om gebruiker linda eigenaar te maken van mijnbestand.

Nadat de eigenaars zijn ingesteld, kunnen permissies toegekend worden. De eenvoudigste wijze om dit te doen is door gebruik te maken van de zogenaamde absolute modus. Hierb wordt elke permissie weergegeven door een getal: 4 staat voor read, 2 voor write en 1 voor execute. Bij gebruik van chmod, geef je 3 getallen als argument: voor de user, group en others entiteit.

Zo gebruik je bijvoorbeeld **chmod 750 mijnbestand** om aan de user read, write en execute te geven (want 4+2+1=7), read en execute aan de groep en niets aan others. Als alternatief kunnen permissies in relatieve modus toegekend worden. Een voorbeeld hiervan is de opdracht **chmod +x mijnbestand**, waarbij mijnbestand voor zowel user, group als others voorzien wordt van de execute permissie.

GEAVANCEERDE PERMISSIES

Omdat de basis permissies niet in alle gevallen voldeden, werden in latere instantie aanvullende permissies toegevoegd. Dit zijn de zogenaamde geavanceerde permissies. Deze permissies worden in heel specifieke gevallen gebruikt (wat betekent dat de kans groot is dat je ze nooit echt nodig zult hebben). Toch is het nuttig een overzicht te hebben van wat ze kunnen doen.

De drie geavanceerde permissies zijn SUID (set user ID) (4), SGID (Set Group ID) (2) en Sticky bit (1). Om ze toe te passen, gebruik je chmod, waarbij een vierde getal gebruikt wordt voor deze permissies en dit getal aan het begin wordt toegevoegd. De opdracht **chmod 4770 mijnfile** zou dus set user ID toevoegen aan het bestand.

De SUID permissie heeft alleen betrekking op bestanden en heeft

geen betekenis voor directories. Deze permissie zorgt ervoor dat -indien toegepast- op programmabestanden, het programma uitgevoerd wordt met de permissies van de eigenaar van het programmabestand en niet met de permissies van de gebruiker die het programma uitvoert. Je ziet deze permissie op enkele systeembestanden, zoals /usr/bin/passwd, dat door middel van SUID kan schrijven in /etc/shadow, een taak die normaal alleen is voorbehouden aan de root user. Als Linux beheerder zou het eigenlijk nooit nodig moeten zijn deze permissie toe te passen.

De SGID permissie kan toegepast worden op bestanden en op directories. Als beheerder van een Linux server, is deze permissie vooral nuttig op directories. De permissie zorgt er namelijk voor dat bestanden die aangemaakt worden in een directory dezelfde groepeigenaar krijgen als de directory zelf. Dat is handig voor gedeelde groepsbestanden.

Stel je voor dat je een groep 'verkoop' hebt, waarvan de leden bestanden met elkaar moeten kunnen delen in /data/verkoop. Als je als lid van de groep verkoop een bestand aanmaakt in deze directory, wordt jouw primaire groep eigenaar van dat bestand en dat is meestal niet wat je wilt. Als je ervoor wilt zorgen dat alle bestanden in /data/verkoop als eigenaar de groep verkoop krijgen (wat ervoor zorgt dat leden van de groep verkoop eenvoudig kunnen wijzigen in elkaars bestanden), gebruik je **chmod g+s /data/verkoop** om de SGID permissie op de directory in te stellen. Vanaf dat moment krijgt elk nieuw bestand in de directory de groep verkoop als groep-eigenaar.

Als laatste speciale permissie is er sticky bit. Deze permissie is erg nuttig op gedeelde directories, waar meerdere gebruikers schrijfrechten hebben. De permissie zorgt er namelijk voor dat een gebruiker alleen bestanden kan verwijderen waarvan de gebruiker eigenaar is of de gebruiker moet eigenaar zijn van de betreffende directory. In een gedeelde directory, zoals de eerder genoemde /data/verkoop waarin leden van de afdeling verkoop schrijfrechten hebben, is dit daarom een waardevolle aanvullende permissie. Gebruikers kunnen alleen bestanden verwijderen die ze zelf aangemaakt hebben. Gebruik **chmod +t /data/verkoop** om op eenvoudige wijze sticky bit toe te voegen.

ACCESS CONTROL LISTS

De laatste toevoeging aan het Linux-systeem van permissies zijn de Access Control Lists. Deze zorgen ervoor dat je specifieke permissies kan geven aan meer dan één gebruiker of aan meer dan één groep. Je past ACLs toe met de opdracht **setfacl** en om te kijken welke ACL's van toepassingen zijn, gebruik je de opdracht **getfacl**. Een voorbeeld is de opdracht **setfacl -m u:linda:rwx /data/verkoop**, waarmee je de gebruiker linda toevoegt als rechthebbende in de directory /data/verkoop, zonder dat ze ingesteld wordt als gebruiker of groepeigenaar van de betreffende directory.

Access control lists kunnen handig zijn, maar het is niet iets wat veel gebruikt wordt. In de meeste gevallen kan je de juiste permissies ook toepassen door middel van groep of user eigenaarschap. Access control lists zijn vooral handig als je een gebruiker of groep toegang wilt geven tot een bepaalde systeemdirectory. Denk bijvoorbeeld aan de web-developer die schrijfrechten moet hebben in de documentroot van de webserver. In dergelijke gevallen zijn access control lists een uitstekende oplossing, omdat je ermee de permissies uitdeelt zonder de bestaande rechtenstructuur te veranderen.