Eindwerk SN: Beveiligingssite

Adriaan Dens (r0257589)
Jan Weyens (s0205128)
Klaas Janssens (r0429246)
Quinten Van Der Auwera (r0258371)

7 maart 2014

Inhoudsopgave

Hoofdstuk 1

Inleiding

In het vijfde semester van de opleiding Toegepaste Informatica aan de Katholieke Hogeschool Leuven krijgt men het vak "Beveiliging". Dit voornamelijk theoretisch vak geeft de studenten een inleiding tot veiligheidsproblemen in de Informatica. Ons eindwerk richt er zich op een plaats te voorzien om deze theorie in de praktijk om te zetten en om informatie te voorzien voor studenten die meer willen weten.

Hoofdstuk 2

Problemen

2.1 Veilig nakijken en uitvoeren van onbekende C code

Voor de oefeningen over exploits in gecompileerde programma's willen we nakijken of de student succesvol het programma heeft kunnen exploiten. Hiervoor laten we toe dat de student zijn C code uploadt via de website waarna het zal nagekeken worden door een intern script.

De vraag is natuurlijk hoe je dit veilig kunt doen. De student kan immers schrijven wat hij wil in het programma dat hij uploadt en uitgevoert zal worden achter de schermen. Oplossingen hiervor zijn *Operating System level Virtualization* of Containers en volledige virtualisatie of Virtuele Machines.

2.1.1 Containers of VMs

Het verschil tussen beiden dat is het verschil tussen *Operating System level Virtualization* en virtuele machines is dat bij een OS-level virtualisatie de containers gebruik maken van de kernel van de host terwijl er bij virtuele machines een strikte scheiding is tussen twee machines, elk gebruikt zijn eigen kernel.

Het voordeel van virtuele machines is daardoor duidelijk, je kan namelijk een Windows VM maken en daarnaast een OS X VM draaien omdat deze toch geen kernel delen. Dit is niet mogelijk met OS-level virtualisatie. Containers hebben dan weer het zeer snel opstart doordat het gebruik maakt van de kernel van het hostsysteem.

Onze voorkeur gaat hierdoor uit naar OS-level virtualisatie omdat we enkel Linux machines nodig hebben en omdat we (zeer) snel een nieuwe container willen kunnen bouwen, gebruiken en dan terug verwijderen.

Er zijn enkele verschillende praktische implementaties van OS-level virtualisatie, ik bespreek hieronder de meest relevante voor onze probleemstelling. Chroots worden bijvoorbeeld niet besproken omdat je hiermee geen limitaties kunt opleggen voor geheugen- en cpugebruik.

LXC Meh. [Graber, 2014].

Docker

BSD Jails

OpenVZ

2.1.2 Praktisch; Hoe gaat het in zijn werk

2.1.3 Conclusie

c code uploaden je kan via radiobuttons selecteren welke opties je aan gcc meegeeft.

Bibliografie

[Graber, 2014] Graber, S. (2014). Userspace tools for the linux kernel containers. Beschikbaar op `http://linuxcontainers.org/`. Bezocht op 15/02/2014.