Visjonsdokument
Linuxbasert dynamisk brannmursystem for nettverk

Espen Gjærde Svein Ove Undal

15.02.2013

1 Revisjonshistorie

Dato	Versjon	Forklaring	Forfattar
01.12.12	1.0	Dokumentet opprettet	Espen, Svein

2 Innleiing

Dette dokumentet er ei analyse av prosjektet som skal gjennomførast, med mål om å avgjere om prosjektet bør startast. Vi vil i dokumentet gå gjennom bakgrunn og mål for prosjektet, greie ut om interessentar og utføre kost/nytte og risikoanalyse. Vi vil også gå gjennom teknologival, standardar for gjennomføring og suksessfaktorar.

3 Bakgrunn for prosjektet

Prosjektet er ei bacheloroppgåve for Espen Gjærde og Svein Ove Undal. Begge studentar ved Avdeling for Informatikk og e-Læring ved Høgskolen i Sør-Trøndelag. Prosjektet går ut på å lage ein smart brannmur basert på pålogging via web. Etter dette skal brukeren få tilpassa reglar for sin bruker.

3.1 Skildring av problem og behov

For administratorar av nettverk med mange brukarar - og mange forskjellige typar brukarar - kan det være vansklig å halde oversikt over nettilgang og effektivt og rettferdig fordle bandbredde og tilgangar mellom brukarene. Spesielt gjeld dette midlertidige nettverk - tildømes dataparty, eller trådløse gjeste-nett. Vi tek sikte på å lage eit system som sikrar rettferdig og dynamisk deling av bandbredde, samstundes som det gir rom for å spesialisere reglar for den enkelte brukar.

3.2 Skildring av dagens system og rutiner

Det finnast i dag døme på system der brukarar må logge inn for å nytte seg av ressursar. Dette er gjerne måten adgangskontroll for trådlausnett vert gjort på på flyplassar og hotell. Dette er det som blir kalla captive-portal[1]. Problemet med slike løysingar er at det enten er fritt fram etter du har logga inn, eller at kapisteten blir delt etter eit "one-size-fits-all"-prinsipp. Systema har og svakhetar som mac-addresse spoofing og dns-tunnel som ein måte å ungå autentiseringa heilt

4 Prosjektmål

Overordna prosjektmål

- lage eit effektivt system som er enkelt å nytte seg av.
- tilegne oss mer kompetanse og erfaring
- lage eit solid og stabilt sluttprodukt

Det er hensiktsmessig å dele måla opp i tre kategoriar:
Effektmål - skildrar oppdagsgiver sine mål med prosjektet.
Resultatmål - skildrar kvar som skal ligger føre når prosjektet er ferdig.
Prossessmål - mål for prosessen i seg sjølv.

[1] Captive-Portal: System som tvingar brukarar via ei bestemt nettside eller gjennom eitt spesielt punkt.

4.1 Effektmål

- Enklere administrajon av brukere i midlertidige nett
- Dynamiske brannmurregler og enkel administrering av desse

4.2 Resultatmål

- Eit fullverdig brannmursystem med innlogging
- Ein fungerande brannmur med individuelle reglar for kvar brukar eller gruppe
- Eit enkelt og oversiktleg administrajonsystem for brannmuren.
- Eit system som sjekkar nettverkskonfigurasjon for pålogga maskiner.
- Ein pakke med enkel installasjon - gjerne .deb eller tar.gz.

4.3 Prosessmål

- Videreutvikle kunnskapar om linux og nettverksikkerhet
- Erfaring med utvikling i python
- Utvikling av eit open source distrubuert system

4.4 Prosjektets omfang

- Det skal vere alternativ å lage informasjon som filer, eller i ein database
- Det ska implementeres ein oversiktlig administrasjonspanel
- Prosjektet skal være ferdigstillit innen 25.mai 0213.
- Systemet blir utvikla og testa for Debian 6.0.6 “wheezy”

4.5 Prosjektets milepælar og hovudaktivitetar

4.5.1 Dokumentasjon

Dato	Mål / milepæl
04.02.2013	Oppstart av prosjekt
15.02.2013	Visjonsdokument i første versjon
10.03.2013	Kravdokument v1
01.04.2013	Arkitekturdokuent v1
25.05.2013	Sluttrapport

Tabell 3.5.1.1: Tabellen viser fristar og kva dokumentasjon som medfølger prosjektet.

4.5.2 Utviklingsteg / Mål

15.02.2013	Fungerende captive-portal
20.02.2013	Brannmur og innlogging styres av python
08.03.2013	Prototype / Testing på TIHLDE-LAN
20.03.2013	IPv6 innført i systemet
01.04.2013	Administrasjonspanel i testversjon
20.04.2013	RC 1 klar.

Tabell 3.5.2.1: Tabellen viser frister for klare mål i utviklingsprosessen.

4.6 Teknologi

Vi vil basere prosjektet på eksisterende teknologi og programmer med åpen kjeldekode. Vi vil utvikle systemet i Debian Linux, men systemet skal kunne kjøre på alle linuxplatformer. Mykje av dei programma vi nyttar finnast også til andre *nix-systemer, og burde med enkle steg kunne nyttast her også. Nedanfor vil vi kort grunngi forskjellige val av teknologi.

Debian Linux “Wheezy”

Debian er den Linuxdistibusjonen med åpen kildekode som er sikrast og mest utbreidd. Det gjere Debian som eit lett valg til vårt prosjekt. I tillegg til dette er Debian den linuxversjonen vi har mest erfaring med.

PHP

For administrasjon og pålogging vil vi nytte webteknologien PHP. Dette er fordi php er et enkelt webspråk som kan kjøres på de fleste platformer og webservere. Det er også et lett språk for en tjener å kjøre.

Python

Som kodespråk mot systemet nyttar vi oss av python. Slik får vi eit abstraksjonsnivå mellom kode og system, som vi ikkje får med bash-script. Dette gjer det også enklare å portere systemet over til andre platformer enn Debian Linux.

ISC Dhcp-server

Dette er ein av dei mest utbredte dhcp-tjenarane i marknaden, og er den klart mest brukte i linux-verda. Vi finn det naturlig å nytte denne både fordi kildekoden er åpen, og fordi det gjer eventuell interasjon mellom systemet vi utvikar og eventuelle eksisterande system enklare.

Iptables

For sjølve brannmuren vil vi nytte iptables. Dette er en enkel pakkefiltrerande brannmur som er støtta i dei fleste *nix-systemer.

OpenLDAP / FreeRadius

Vi vil kjøre autentisering gjennom Linux PAM[2]. Dette gjer at sluttbrukar står fritt til å velje brukerdatabase. For demonstrasjon og testing vil vi bruke OpenLDAP og FreeRadius som brukardatabasar.

[2] Pluggable Authentiction Modules: Eit fleksibelt authentiseringsystem. Gir mulighet for mange forskjellige typar brukardatabasar.

5 Interessenter og rammebetingelser

5.1 Interessentanalyse

Interessent	Suksesskriterie	Bidrag til prosjektet
Eksterne interessenter		
Høgskolen i Sør-Trøndelag		Oppdragsgiver
TIHLDE	Vellykket test under LAN	TIHLDE-LAN, testlab
Sluttbruker	Enkelt og effektivt system	Kritikk, bruk, spreding av sluttprodukt
Interne interessenter		
Studentane		Systemutviklarar - prosjekeigarar

Veileder		Kunnskap, Vegleiing

Tabell 4.1.1: Oversikt over interessentar

5.2 4.2 Rammebetingelser

- Produkt og dokumentasjon ferdig innan 25.mai 2013
- Produktet skal kunne installerast og brukast av ein brukar utan store krav til linux/nettverks kunnskap.
- Systemet er utvikla og testa i Debian 6.0.6 "Wheezy"
- Utviklingsmetoden Unified Process vert nytta.
- Systemet vert basert på og utgitt som åpen kildekode.

6 5 Kritiske suksessfaktorer

6.1 5.1 Suksessfaktorer

Sømløst system, som fungerer uten at sluttbrukar merker at systemet er der. Systemet skal effektivt gjere nettverket bedre å bruke med tanke på ping og ytelse under stress.

Systemet skal vere særdeles lett å settast opp, det skal ikkje krevast store kunnskapar innan linux og nettverksadministrasjon for å få systemet til å fungere.

6.2 5.2 Informasjonsbehov

- Veileder skal haldast oppdatert om framgangen i prosjektet

7 6 Risikoanalyse

Vi har identifisert følgande risikoar i prosjektet

A. Langvarig sjukdom/fravær av gruppemedlemmar
B. Samarbeidsvanskar i teamet
C. Feil på programvare vi er avhengig av

D. Produkt ikkje i kjørbar versjon ved testtid
E. Programvare oppfører seg ikkje som forutsett
F. Vanskeleg brukargrensesnitt
G. For høge systemkrav
H. Tap av viktig data

Sjå Vedlegg A for risikoscore og utrekning av sannsyn og konsekvens
[Warning: Image ignored]
Figur 6.1: Viser risikoplassering av dei kartlagte risikoane i prosjektet.

8 7 Kost/nytte-analyse

8.1 7.1 Kvantifiserbar og ikkje-kvantifiserbar nytte

Dette er eit bachelorprosjekt som baserar seg på bruk av åpen kildekode og alle nytta verktøy er åpne og fritt tilgjengeleg. Vi vil også publisere all kildekode under ein fri lisens. Det er derfor vanskleg å sette opp gode kostnadsvurderingar; arbeidskraft, materia og verktøy er gratis.

8.1.1 7.1.1 Kvantifiserbar nytte

- Mindre administrajonsarbeid for nettverksansvarleg

8.1.2 7.1.2 Ikkje-kvantifiserbar nytte

- Betre fordeling av bandbredde

8.2 7.2 Bortfall av direkte kostnadar

8.3 7.3 Estimerte kostnadar

- Prosjektgruppa består av to studentar som kvar skal arbeide omlag 450 timar
- Veiledar har 30 timar til disposisjon for vegleiing
- Drift og vedlikehold blir små eller uvensentlige kostnadar. Med tanke på maskinvarekrav skal dette systemet ha svært lave krav. Einaste maskinvare krav er to nettverksportar.

8.4 7.4 Samanstilling av kostnadar og nytte

Det har i dette prosjektet vore vansklig å lage ei klar vurdering av kost og nytte. Vi har få klare kvantifiserbare nytteeffektar, og einaste klare kostnaden timestallet som skal brukast. Dette timetallet er igjen ikkje spesielt for prosjektet, men ein del av eikvar bacheloroppgåve.

9 8 Retningslinjer og standardar

9.1 8.1 Krav til dokumentasjon

Følgande dokumentasjon skal lagast:

- Visjonsdokument
- Kravdokument
- Arkitekturdokument
- Brukerdokumentasjon
- Installasjonsvegleiing
- Sluttrapport

9.2 8.2 Krav til kvalitetskontroll

Kvalitets gjennomgang blir tildels dekket av de daglige utviklingsmøtene i prosjektgruppa. Vi får her tilbakemelding om eventuelle endringer som burde vært endret i systemet, og om det må tilføyes noe.

Prosjektgruppa må også ha en gjennom gang av følgende:

- Python kode
- PHP/HTML kode
- Brukervennlighet på web siden
- Databasestruktur og sikkerhet

I tillegg til å kvalitetssikre, tester vi programvaren underveis og etter kvar endring. Testingen underveis blir en peikepinne på om systemet fungerer som det skal, samt at vi utbedrer alle problemer når de oppstår. Vi vil òg ha ei utbredt testing av ein prototype på TIHLDE-lan, som vil effektivt vise svakheter og styrker i systemet.

9.3 8.3 Krav til standardar og metoder

Systemet skal bruke kjent teknologi og basere seg på innførte standardar. Dette fordi det for sluttbrukar sine klientar ikkje skal være behov for spesialutstyr eller eigne klientprogram.

9.3.1 8.3.1 Programmeringstandardar

- Programmering skjer i språka HTML, PHP og python.
- Alle metodar skal ha engelske navn, og følge kjente namnekonvensjonar i sine respektive språk.
- Kommentarar og forklaringar i kodefilene skal skrivast på engelsk.

9.3.2 8.3.2 Konfigurasjonsfiler

- Konfigurasjonsfiler og programvare skal i størst mulig grad følge standardar og ligge der det er naturlig i eit linux / Debian-system.
- Kommentarar og forklaringar i filene skal skrivast på engelsk

9.3.3 8.3.3 Bruk av verktøy

- Versjonskontroll skjer med versjonshandteringssystemet GIT.
- Systemet blir testa på ei tjenermaskin med operativsystemet Debian 6.0.6 “Wheezy”.
- evt. mySQL?

9.3.4 8.3.4 Andre standardar

- Systemet skal baserast på kjente og de-facto standard protokollane IP, TCP, DHCP og HTTP.
- Språk for administrajonsystemet skal være norsk - nynorsk.

9.4 8.4 Endringshandtering

Vi har god praktisk kunnskap og erfaring med de forskjellige teknologiene som inngår i linux, men som eit konstant forandrandes opensource-system kan det oppstå att vi må endre på nokon av teknologivala og også funksjoner i systemet. Vi vil derfor lage nokre enkle retningslinjer for korleis vi skal behandle endringer i systemet.

Retningslinjer ved systemendringar

1. Eventuell mistanke om problemer meldast tidleg til dei andre i prosjektgruppa
2. Det skal arbeidast med å få oversikt over endringar og ringverknadar.
3. Endringer dokumenterast og avvik rapporterast
4. Tidsplaner justerast
5. Endringer gjennomførast
6. Evaluering av endring og endringsprosessen vert gjennomført

10 9 Prosjektorganisering

[Warning: Image ignored]

11 10 Tilråding om vidare arbeid

Med bakgrunn i forstudiearbeidet og dette visjonsdokumentet vert det tilrådd at prosjektet vert utvikla vidare.

12 11 Vedlegg

Vedlegg A	Risikoanalyse