Defending Active Directory Trusts

Active Directory(AD)

ယနေ့ခေတ် လုပ်ငန်းအဖွဲ့အစည်းတွေမှာ တစ်ခုတည်းသော ကွန်ပျူတာစနစ်၊ တစ်ခုတည်းသော Network နဲ့ ပဲ အလုပ်လုပ်နေကြတာ မဟုတ်တော့ပါဘူး။ ကုမ္ပဏီတစ်ခုထဲမှာတင် မဟုတ်ဘဲ မတူညီတဲ့ ဌာနတွေ၊ မ တူညီတဲ့ Domain တွေ၊ တစ်ခါတရံမှာ အပြင်ဘက် Partner ကုမ္ပဏီတွေနဲ့ပါ ချိတ်ဆက်ပြီး အလုပ်လုပ်ရတဲ့ အခြေအနေတွေ များလာပါတယ်။ ဒီလို အခြေအနေတွေမှာ User Account တွေ၊ Permission တွေ၊ Resource တွေကို စနစ်တကျ စီမံခန့်ခွဲနိုင်ဖို့ Active Directory ကို အသုံးပြုကြပါတယ်။

Trust

Active Directory Trust ဆိုတာက Domain တစ်ခုနဲ့ Domain တစ်ခုကြားမှာ “တစ်ဖက်က အခြားဖက်ကို ယုံကြည်ထားတယ်” လို့ သဘောတူထားတဲ့ ဆက်သွယ်ရေး စနစ်တစ်ခုလိုပါပဲ။ ဥပမာ Domain A မှာရှိတဲ့ User တစ်ယောက်က Domain B ထဲက Server သို့မဟုတ် File Share ကို အသုံးပြုနိုင်ဖို့ Trust တစ်ခု ချိတ်ထားရပါမယ်။ ဒီ Trust ရှိတဲ့အတွက် User အသစ်တစ်ယောက်ကို မထပ်ခါထပ်ခါ ဖန်တီးစရာမလိုဘဲ အလုပ်လုပ်နိုင်တာကြောင့် လုပ်ငန်းအတွက် အဆင်ပြေပါတယ်။ ဒါကြောင့် Collaborative Business Environment တွေမှာ Trust တွေဟာ အလွန်အရေးကြီးတဲ့ အခန်းကဏ္ဍကို ထမ်းဆောင်နေပါတယ်။

Risk

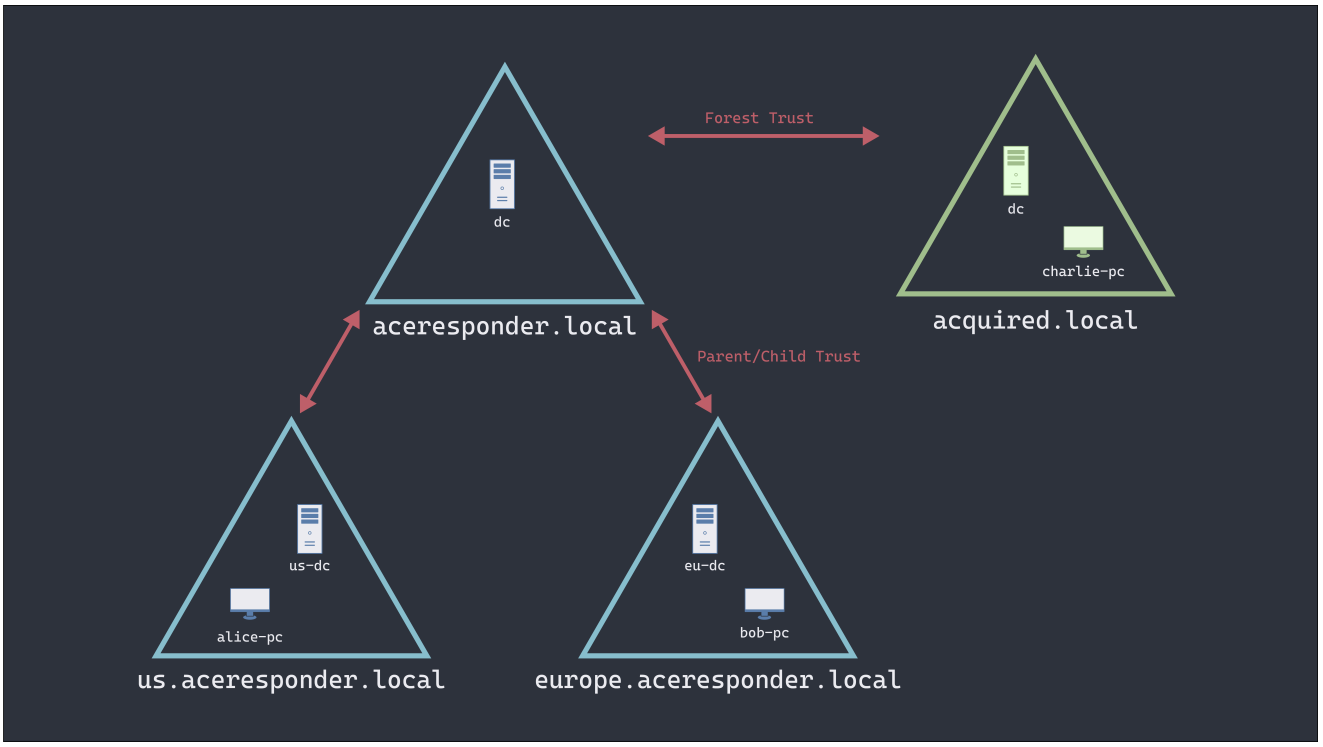
ဒါပေမယ့် Trust တွေဟာ အကျိုးကျေးဇူးပဲ မဟုတ်ဘဲ အန္တရာယ်တွေပါ ပါဝင်လာပါတယ်။ Active Directory ကိုယ်တိုင်က စနစ်ကြီးတစ်ခု ဖြစ်တာကြောင့် မသေချာရင် Vulnerability တွေ ရှိနိုင်သလို Trust တွေမှာလည်း ထူးခြားတဲ့ Security ပြဿနာတွေ ရှိပါတယ်။ Trust တစ်ခု ချိတ်လိုက်တာက Domain တစ် ခုရဲ့ လုံခြုံရေးကို အခြား Domain တစ်ခုနဲ့ ချိတ်ဆက်လိုက်တာနဲ့ တူပါတယ်။ အခြား Domain ဘက်မှာ အားနည်းချက်ရှိနေရင် အဲဒီ အားနည်းချက်ကို အသုံးချပြီး ကိုယ့် Domain ထဲကိုပါ ဝင်လာနိုင်တဲ့ လမ်းကြောင်း ဖြစ်သွားနိုင်ပါတယ်။

အဖွဲ့အစည်းအများစုက Trust ရှိတယ်ဆိုတာကို သိပေမယ့် အဲဒီ Trust က ဘယ်လောက်အထိ အန္တရာယ်ရှိ နိုင်လဲ၊ ဘယ်လို Attack Path တွေ ဖြစ်လာနိုင်လဲ ဆိုတာကို မသိကြတာ များပါတယ်။ အထူးသဖြင့် အရင် ကတည်းက ချိတ်ထားပြီး မကြာခဏ ပြန်စစ်မကြည့်တော့တဲ့ Trust တွေဟာ Attacker အတွက် အခွင့်အရေးကောင်း ဖြစ်လာနိုင်ပါတယ်။ Domain တစ်ခုထဲမှာ Credential တစ်ခုရသွားတာနဲ့ Trust ကို အသုံးချပြီး အခြား Domain ထဲကို ရောက်သွားနိုင်တဲ့ အခြေအနေတွေ ဖြစ်နိုင်ပါတယ်။

Active Directory Trusts

Trust ကို အသုံးပြုတဲ့ အကြောင်းရင်းတွေထဲမှာ အရေးကြီးဆုံးတစ်ခုက Administration ကို ခွဲခြားတာ ဖြစ် ပါတယ်။ Organization တစ်ခုမှာ Enterprise-level Administrator တွေ ရှိပြီး Organization အားလုံးကို စီမံခန့်ခွဲနိုင်ပါတယ်။ တစ်ဖက်မှာလည်း Region သို့မဟုတ် Site တစ်ခုချင်းစီကိုသာ စီမံခန့်ခွဲတဲ့ Regional Administrator တွေ ရှိနိုင်ပါတယ်။ Domain ကို ခွဲပြီး Trust ချိတ်ထားတဲ့အတွက် Authority ကို ချဲ့ထွင်မ သွားဘဲ တာဝန်ခွဲခြား စီမံနိုင်တာ ဖြစ်ပါတယ်။ တစ်ခါတရံမှာတော့ အခြား Organization တစ်ခုကို ယုံကြည်

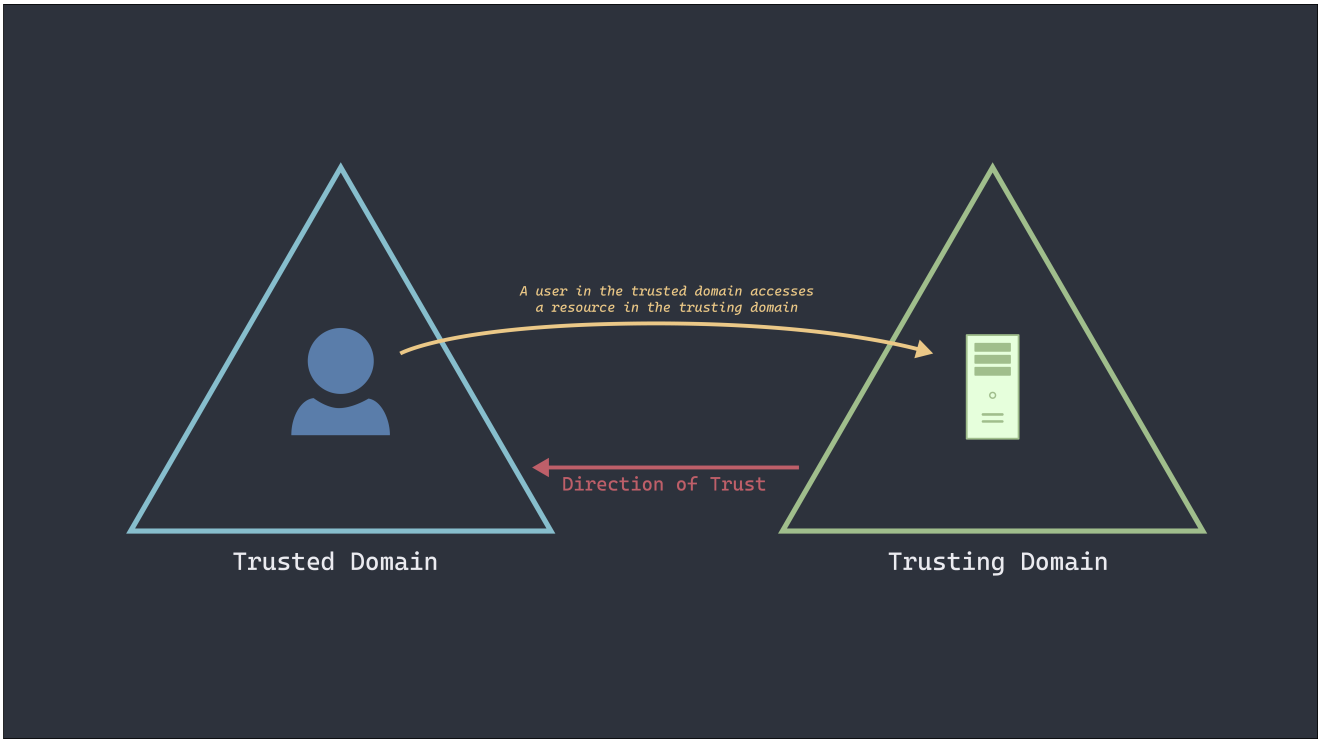
ပြီး သူတို့ရဲ့ User တွေကို ကိုယ့် Forest ထဲက Resource တွေကို Authenticate လုပ်ခွင့်ပေးတဲ့ အခြေအနေတွေပါ ရှိပါတယ်။



User တစ်ယောက်က သူ့ Domain မဟုတ်တဲ့ အခြား Domain ထဲက Resource တစ်ခုကို Access လုပ်ချင်တဲ့အခါ Trust Relationship က အဓိကအခန်းကဏ္ဍ ထမ်းဆောင်ပါတယ်။

> Resource ပါရှိတဲ့ Domain ကို Trusting Domain လို့ ခေါ် ပြီး User Account ပါရှိတဲ့ Domain ကို Trusted Domain လို့ ခေါ်ပါတယ်။

Trust ချိတ်ထားပြီး User ကို Permission ပေးထားရင် User က Trusting Domain ထဲမှာ Account အသစ်မဖန်တီးဘဲ Access လုပ်နိုင်ပါတယ်။ ဒီအရာက လုပ်ငန်းအတွက် အဆင်ပြေစေသလို Management ကိုလည်း လွယ်ကူစေပါတယ်။



ဒါပေမယ့် Resource ကို တကယ် Access လုပ်ဖို့ဆိုရင် User က Kerberos TGS Ticket ကို Resource ရှိတဲ့ Domain Controller ကနေ ရယူရပါမယ်။ ဒီ Ticket ကို ရယူဖို့ Trust Path ဆိုတဲ့ Authentication လမ်းကြောင်းကို ဖြတ်သန်းရပါတယ်။

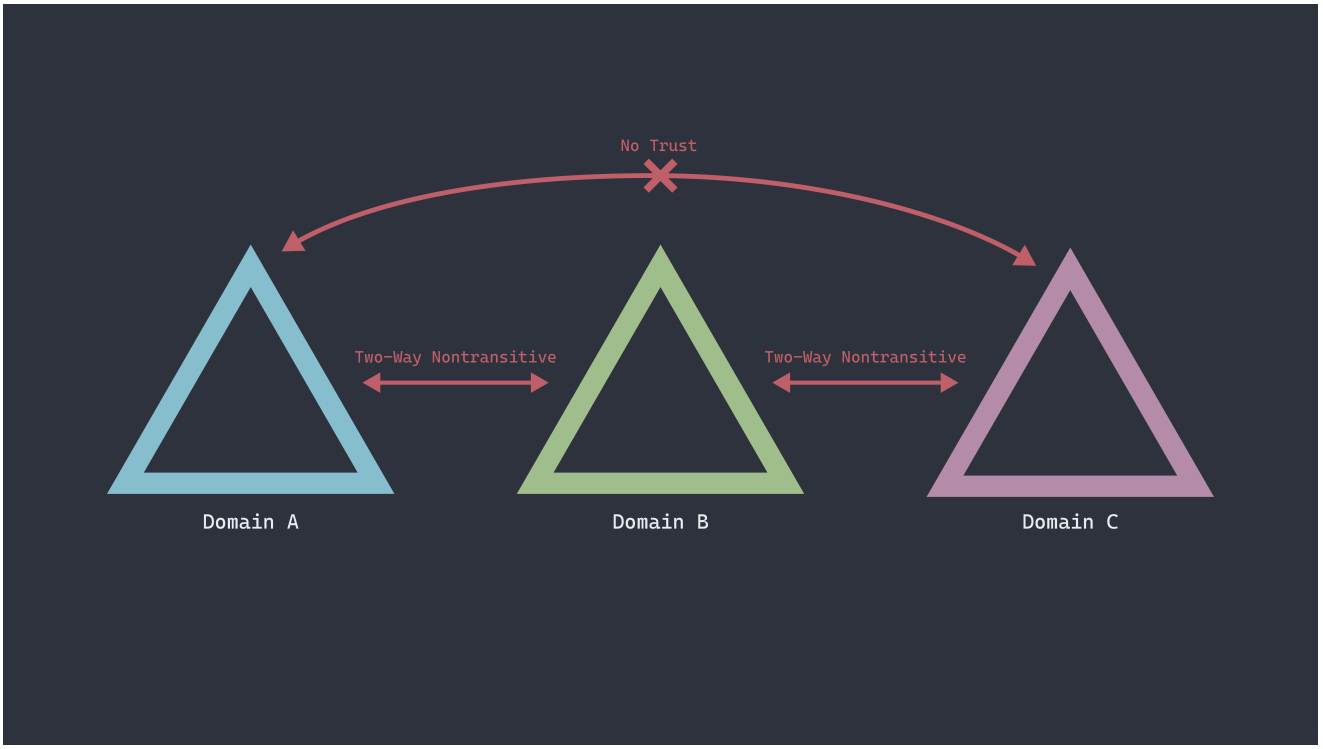
> Trust Path ဆိုတာ User ရဲ့ Domain ကနေ စပြီး Resource ရှိတဲ့ Domain အထိ Authentication လုပ်ရမယ့် Domain တွေအားလုံးကို ဆိုလိုပါတယ်။

User ရဲ့ Domain နဲ့ Resource Domain ကြားမှာ Direct Trust ရှိရင် လွယ်ကူပါတယ်။ ဒါပေမယ့် အလယ်မှာ Domain အချို့ပါဝင်နေရင် Domain Controller တစ်ခုချင်းစီက Request ကို စစ်ပြီး နောက်ထပ် Domain ကို Referral ပေးရင်း နောက်ဆုံး Resource Domain ထိ ရောက်သွားပါတယ်။

ဒါကို နားလည်ဖို့ အရေးကြီးတဲ့ အချက်တစ်ခုက Trust ရှိတယ်ဆိုတိုင်း Trust Path အမြဲရှိနေမယ်လို့ မထင်သင့်ပါဘူး။ Trust က Transitive ဖြစ်ရပါမယ်။

> Transitive Trust ဆိုတာ Trust ကို အခြား Domain တွေအထိ ဆက်လက် အသုံးပြုခွင့်ပေးတာ ဖြစ်ပါတယ်။ Non-transitive Trust ဖြစ်ရင် Domain နှစ်ခုကြားမှာပဲ အလုပ်လုပ်ပြီး အခြား Domain တွေကို မဆက်သွယ်နိုင်ပါဘူး။

ဒါကြောင့် Trust Path တစ်ခု တည်ဆောက်နိုင်မနိုင် ဆိုတာက Transitivity ပေါ်မှာ အလွန်မူတည်ပါတယ်။



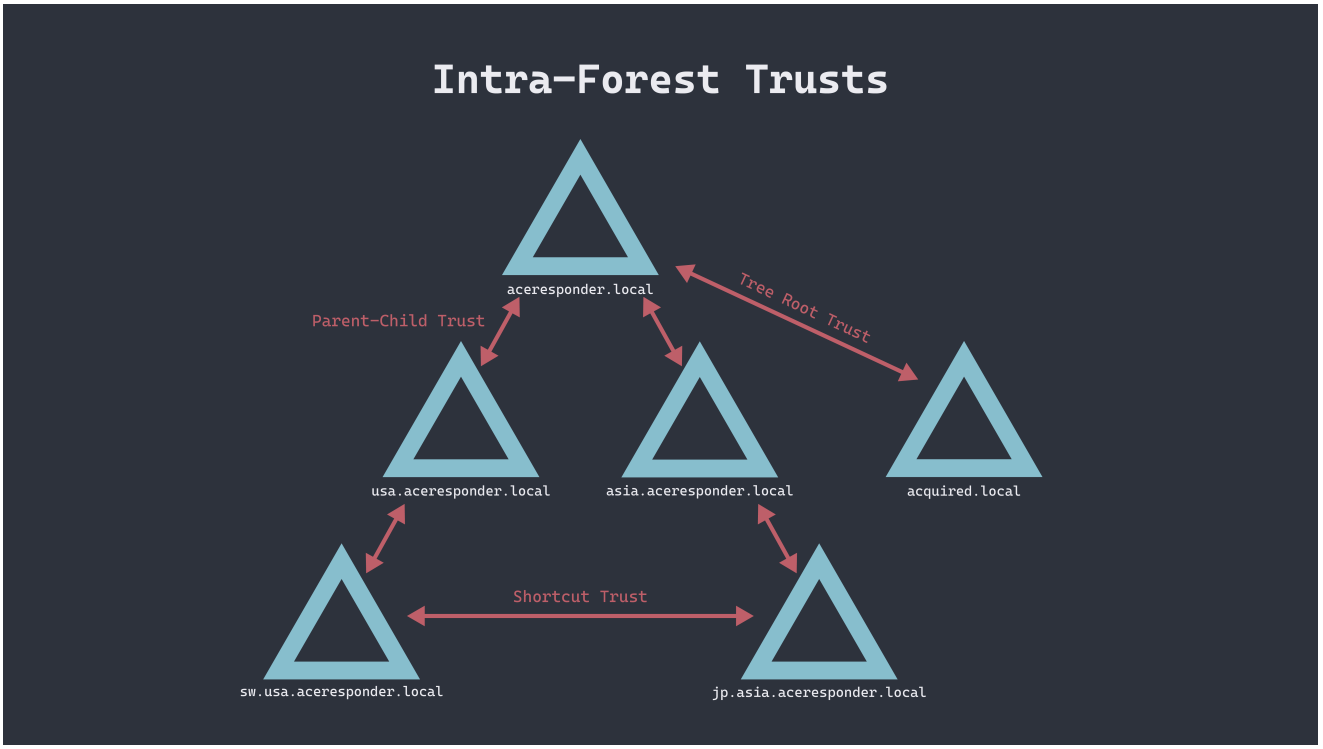
Trust Relationship တွေမှာ One-way နဲ့ Two-way ဆိုပြီး ကွာခြားမှုရှိပါတယ်။ Two-way Trust ဆိုရင် Domain နှစ်ခုစလုံးက User တွေ အပြန်အလှန် Access လုပ်နိုင်ပါတယ်။ One-way Trust မှာတော့ Trusted Domain က User တွေက Trusting Domain ထဲကို Access လုပ်နိုင်ပေမယ့် Trusting Domain က User တွေက ပြန်ပြီး Trusted Domain ထဲကို မဝင်နိုင်ပါဘူး။ One-way trust က အသွားရှိပြီး အပြန်မရှိပါဘူး။ ဒီကွာခြားမှုက Security Design အတွက် အရေးကြီးပါတယ်။

Trust Type

Defender အနေနဲ့ အထူးသတိထားရမယ့် အချက်က Intra-forest Trust နဲ့ Inter-forest Trust ကွာခြားမှု ဖြစ်ပါတယ်။ **Intra-forest Trust** ဆိုတာ Forest တစ်ခုတည်းအတွင်းမှာရှိတဲ့ Domain တွေကြား Trust

ဖြစ်ပြီး Default အနေနဲ့ Two-way ဖြစ်ပြီး Transitive လည်း ဖြစ်ပါတယ်။

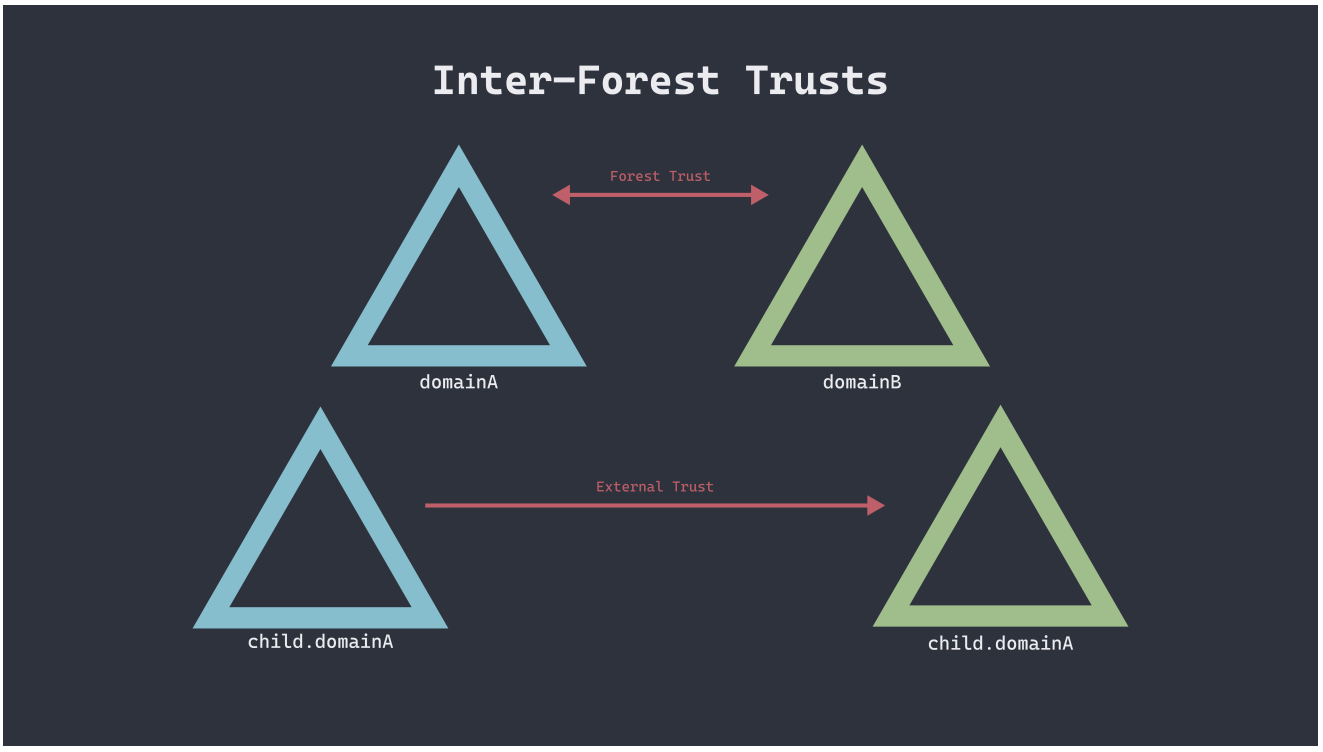
- Domain Tree အသစ်ထည့်တဲ့အခါ **Tree-root Trust** ဖြစ်လာပြီး
- Child Domain အသစ်ထည့်ရင် **Parent-child Trust** ဖြစ်လာပါတယ်။
- Forest ကြီးပြီး Trust Path ရှည်လွန်းတဲ့အခါ **Shortcut Trust** ကို အသုံးပြုပြီး Authentication ကို မြန်စေပါတယ်။



Inter-forest Trust ဆိုတာတော့ Forest မတူတဲ့ Domain တွေကြား Trust ဖြစ်ပါတယ်။

- **Forest Trust** က Forest နှစ်ခုလုံးကို ချိတ်ဆက်ထားပြီး
- **External Trust** ကတော့ Forest မတူတဲ့ Domain နှစ်ခုကြား Non-transitive Trust ဖြစ်ပါတယ်။

Security အနေနဲ့ ကြည့်ရင် AD ရဲ့အဓိက Security Boundary က Forest ဖြစ်ပါတယ်။ Forest အတွင်းမှာ Privileged Account တစ်ခု ရှိနေပြီဆိုရင် Domain Boundary တွေကို ကျော်နိုင်တဲ့ စွမ်းအား ရှိနေပါတယ်။ ဒါက Design အရ ဖြစ်တာကြောင့် အန္တရာယ်ကို လျော့ချနိုင်ပေမယ့် လုံးဝ ပယ်ဖျက်ဖို့တော့ ခက်ခဲပါတယ်။



ဒါကြောင့် Domain ကို ခွဲထားတာက Access Control နဲ့ Risk Reduction အတွက် အသုံးဝင်ပေမယ့် Child Domain တစ်ခုကို Attacker က အပြည့်အဝ ထိန်းချုပ်နိုင်သွားရင် နောက်ဆုံးမှာ Enterprise Level အထိ Escalate လုပ်နိုင်မယ်လို့ မျှော်မှန်းထားရပါမယ်။ ဒီသဘောတရားကို နားလည်ထားခြင်းက Active Directory Trusts ကို ကာကွယ်တဲ့အခါ အလွန်အရေးကြီးပါတယ်။

Trust Authentication

User တစ်ယောက်က Domain တစ်ခုထဲမှာ Log in လုပ်တဲ့အချိန် Domain Controller က User ကို Kerberos Authentication သုံးပြီး အတည်ပြုပါတယ်။ ဒီအချိန်မှာ Domain Controller က User အတွက် TGT (Ticket Granting Ticket) တစ်ခု ထုတ်ပေးပါတယ်။ ဒီ TGT ကို krbtgt ဆိုတဲ့ Account ရဲ့ Password Hash နဲ့ Encrypt လုပ်ပြီး Sign လုပ်ထားပါတယ်။ krbtgt Account ဆိုတာ Domain တစ်ခုချင်းစီမှာ မဖြစ်မနေရှိတဲ့ အရေးကြီးတဲ့ Account ဖြစ်ပြီး Domain တိုင်းမှာ krbtgt Hash က မတူပါဘူး။ အဲ့ဒါကြောင့် Domain တစ်ခုမှာ ထုတ်ထားတဲ့ TGT ကို အခြား Domain တစ်ခုမှာ တိုက်ရိုက် အသုံးမပြုနိုင်ပါဘူး။

User က Service တစ်ခုကို Access လုပ်ချင်တဲ့အခါ Domain Controller က User ပေးလာတဲ့ TGT ကို krbtgt Hash နဲ့ စစ်ဆေးပြီး မှန်ကန်ရင် Service Ticket (TGS) ကို ထုတ်ပေးပါတယ်။ ဒါက Domain တစ်ခုအတွင်းမှာ ဖြစ်တဲ့ Normal Authentication Flow ဖြစ်ပါတယ်။ ဒါပေမယ့် Resource က အခြား Domain ထဲမှာ ရှိနေရင် ဒီပုံစံနဲ့ မလုံလောက်တော့ပါဘူး။

Inter-realm Trust Key

Trust တစ်ခုကို Domain နှစ်ခုကြားမှာ ဖန်တီးလိုက်တဲ့အခါ Domain နှစ်ဖက်လုံးမှာ Trust Account တစ်ခုစီ အလိုအလျောက် ဖန်တီးပါတယ်။ ဒီ Trust Account ကို Trusting Domain ရဲ့ Domain Name နောက်မှာ $ တစ်ခု ထည့်ပြီး အမည်ပေးထားပါတယ်။ ဥပမာ aceresponder.local ကို Trusting Domain လို့ သတ်မှတ်ရင် aceresponder$ ဆိုတဲ့ Trust Account တစ်ခု ဖြစ်လာပါမယ်။ ဒီ Trust Account တွေရဲ့ Password Hash တွေကို အသုံးပြုပြီး Inter-realm Trust Key ဆိုတာကို ဖန်တီးပါတယ်။

User တစ်ယောက်က သူ့ Domain မဟုတ်တဲ့ အခြား Domain ထဲက Resource ကို Access လုပ်ဖို့ ကြိုးစားတဲ့အခါ Domain Controller က Inter-realm TGT ဆိုတဲ့ အထူး Ticket တစ်မျိုးကို ထုတ်ပေးပါတယ်။ ဒီ Inter-realm TGT ကို krbtgt Hash နဲ့ Encrypt မလုပ်ဘဲ Inter-realm Trust Key နဲ့ Encrypt လုပ်ထားပါတယ်။ နောက် Domain က Domain Controller က ဒီ Ticket ကို လက်ခံရရှိတဲ့အခါ သူ့မှာရှိတဲ့ Inter-realm Trust Key နဲ့ Decrypt လုပ်ပြီး စစ်ဆေးပါတယ်။ အရာအားလုံး မှန်ကန်နေတယ်ဆိုရင် နောက်ဆုံး Resource အတွက် Service Ticket ကို ထုတ်ပေးပါတယ်။

ဒီသဘောတရားကို နားလည်ဖို့ အောက်ပါ လက်တွေ့ဥပမာကို ကြည့်ရင် ပိုရှင်းလင်းလာပါမယ်။ ဒီ Scenario မှာ Parent Domain တစ်ခု ဖြစ်တဲ့ aceresponder.local ရှိပြီး Child Domain နှစ်ခု ဖြစ်တဲ့ us.aceresponder.local နဲ့ eu.aceresponder.local ရှိပါတယ်။ US Domain ထဲက User တစ်ယောက် က EU Domain ထဲက Resource တစ်ခုကို Access လုပ်ချင်တယ်ဆိုရင် Trust Path ကို လိုက်ပြီး Authentication လုပ်ရပါမယ်။

Timestamp	EventCode	EventAction	HostHostnam	ServiceName	TargetUserName
2023-10-20 15:49:50.107	4769	Kerberos Service Ticket Operations	eu-dc	BOB-PC$	alice@US.ACERESPONDER.LO
2023-10-20 15:49:50.101	4769	Kerberos Service Ticket Operations	dc	EU.ACERESPONDER.LOCAL	alice@US.ACERESPONDER.LO
2023-10-20 15:49:50.099	4769	Kerberos Service Ticket Operations	us-dc	ACERESPONDER.LOCAL	alice@US.ACERESPONDER.LO
2023-10-20 15:49:50.096	4768	Kerberos Authentication Service	us-dc	krbtgt	alice

Rows per page 10 1-4 of 4

US Domain ထဲက alice ဆိုတဲ့ User က bob-pc ဆိုတဲ့ EU Domain ထဲက Computer ကို Access လုပ်တဲ့အချိန် US Domain Controller က alice ကို TGT တစ်ခု ပေးပါတယ်။ နောက်တစ်ဆင့်မှာ US Domain Controller က aceresponder.local နဲ့ Trust ရှိနေတဲ့အတွက် aceresponder$ Trust Account ကနေ ထုတ်ထားတဲ့ Trust Key ကို အသုံးပြုပြီး Inter-realm TGT တစ်ခု ဖန်တီးပြီး Referral အနေနဲ့ ပြန်ပေးပါတယ်။ အဲဒီ Ticket ကို Parent Domain Controller က လက်ခံပြီး EU Domain နဲ့ Trust ရှိနေတဲ့ EU$ Trust Account ကို အသုံးပြုပြီး နောက်ထပ် Inter-realm TGT တစ်ခု ဖန်တီးပါတယ်။ နောက်ဆုံးမှာ EU Domain Controller က ဒီ Ticket ကို စစ်ဆေးပြီး bob-pc အတွက် Service Ticket ကို ထုတ်ပေးပါတယ်။ Service Ticket ကို ရပြီးတဲ့နောက်မှာ alice က bob-pc ကို အောင်မြင်စွာ Log on လုပ်နိုင်သွားပါတယ်။

Event Log အနေနဲ့ ကြည့်ရင် Alice က US Domain Controller ကနေ TGT ရတဲ့အခါ Event ID 4768 ကို မြင်ရပြီး Trust Path တစ်လျှောက် Inter-realm TGT တွေ ထုတ်ပေးတဲ့အချိန် Event ID 4769 တွေ အစဉ်လိုက် ပေါ်လာပါတယ်။ နောက်ဆုံးမှာ Computer ထဲကို Log on ဝင်သွားတဲ့အခါ Event ID 4624 ကို မြင်ရပါမယ်။ ဒီ Event Sequence က Trust Authentication အလုပ်လုပ်ပုံကို လက်တွေ့ပြသပေးတဲ့ ဥပမာတစ်ခု ဖြစ်ပါတယ်။

The sequence of events in order:

1. **4768** - Alice receives a TGT from US-DC, the US domain controller
2. **4769** - US-DC creates an inter-realm TGT with the trust key derived from the aceresponder$ password and returns a referral.
3. **4769** - DC, the domain controller for aceresponder.local creates another inter-realm TGT with the trust key derived from the EU$ password and returns a referral.

4. **4769** - EU-DC, the domain controller for EU validates the TGT and returns a service ticket for BOB-PC$.
5. **4624** - *alice* logs on to *bob-pc*.။

Trust Enumeration

US Domain ထဲက alice ဆိုတဲ့ User တစ်ယောက်က EU Domain ထဲမှာရှိတဲ့ bob-pc ကို Access လုပ်နိုင်ဖို့ဆိုရင် Trust ရှိနေတာတစ်ခုတည်းနဲ့ မလုံလောက်ပါဘူး။ alice ကို bob-pc ပေါ်မှာ သင့်တော်တဲ့ Local Group ထဲ ထည့်ထားရပါမယ်။ ဥပမာ Remote Desktop Users သို့မဟုတ် Administrators လို Group တွေ ဖြစ်နိုင်ပါတယ်။ ဒါ့အပြင် Resource တွေပေါ်မှာ သတ်မှတ်ထားတဲ့ DACLs (Discretionary Access Control Lists) တွေကနေတဆင့် Access ပေးထားတာလည်း ဖြစ်နိုင်ပါတယ်။ တစ်ခါတရံမှာတော့ Foreign Domain က Group တွေထဲကို User ကို တိုက်ရိုက် ထည့်ထားတာလည်း တွေ့ရပါတယ်။

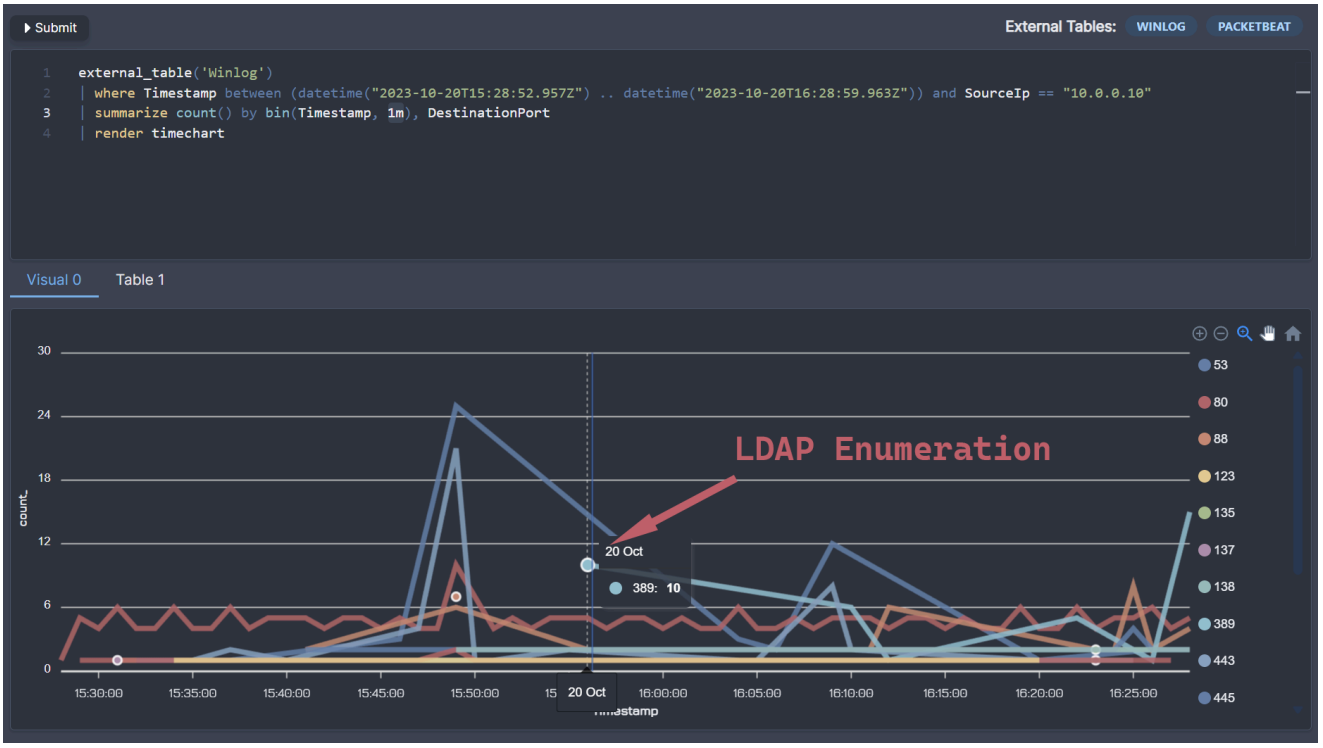
ဒီနေရာမှာ အန္တရာယ်များတဲ့ Misconfiguration တွေ အများကြီး ဖြစ်ပေါ်တတ်ပါတယ်။ Individual User တွေကို DACL Entry တစ်ခုချင်းစီနဲ့ စီမံရတာက အလွန်ခက်ခဲပြီး အမှားလုပ်မိနိုင်ခြေ မြင့်ပါတယ်။ အရင်က Active Directory DACL Abuse အကြောင်းကို လေ့လာဖူးသူတွေဆိုရင် Permission တစ်ခုပဲ မှားသွားရင် Attacker အတွက် Path တစ်ခု ဖွင့်ပေးလိုက်သလို ဖြစ်သွားနိုင်တာကို နားလည်ကြမှာပါ။ Trust ရှိတဲ့ အခြေအနေမှာ ဒီလို Misconfiguration တွေက Domain တစ်ခုထဲမှာသာ မကဘဲ Domain အခြားတစ်ခုအထိ အကျိုးသက်ရောက်သွားနိုင်ပါတယ်။

Attackers တွေအနေနဲ့ Trust Enumeration ကို အများအားဖြင့် Domain Enumeration လုပ်တဲ့အဆင့်ထဲမှာ တစ်ခါတည်း လုပ်သွားကြပါတယ်။ သူတို့ရဲ့ ရည်ရွယ်ချက်က

- Domain တွေဘယ်လို ချိတ်ဆက်ထားလဲ၊
- Trust Path ဘယ်လို ရှိလဲ၊
- ဘယ် Domain က ဘယ် Domain ကို ယုံကြည်ထားလဲ ဆိုတာကို

အပြည့်အဝ နားလည်ဖို့ ဖြစ်ပါတယ်။ ဒီအတွက် အသုံးအများဆုံး Tool က **SharpHound** ဖြစ်ပါတယ်။ SharpHound က BloodHound အတွက် Data စုဆောင်းပေးတဲ့ Tool ဖြစ်ပြီး Trust Information တွေကို အလွန်လွယ်ကူစွာ စုဆောင်းနိုင်ပါတယ်။

Defender အနေနဲ့ ဒီ Enumeration Activity ကို Detect လုပ်ဖို့ဆိုတာ အလွန်မလွယ်ကူပါဘူး။ ဘာကြောင့်လဲဆိုတော့ Trust Enumeration လုပ်တဲ့ Traffic တွေဟာ Normal AD Operation တွေနဲ့ အရမ်းတူပါတယ်။ ဥပမာ SharpHound က Trust Enumeration လုပ်တဲ့အချိန် LDAP Port ဖြစ်တဲ့ 389 ကို အသုံးပြုပြီး Query တွေ ပို့ပါတယ်။ ဒီ Query တွေဟာ Admin Tool တွေ သုံးတဲ့ ပုံစံနဲ့ များစွာ ဆင်တူနေပါတယ်။



ပေးထားတဲ့ Scenario ထဲမှာ Attacker က SharpHound ရဲ့ trusts collection method ကို သုံးပြီး Domain Trust တွေကို စုဆောင်းပါတယ်။ Log တွေကို ကြည့်ရင် LDAP Traffic ရဲ့ အရေအတွက်က Baseline နဲ့ အရမ်းမကွာခြားပါဘူး။ ထူးထူးခြားခြား Spike ကြီးတစ်ခု မတွေ့ရတဲ့အတွက် “ဒါက တိုက်ခိုက်မှုပါ” လို့ သတ်မှတ်ဖို့ ခက်ခဲပါတယ်။ ဒီလို Pattern က Host အများကြီးမှာ အမြဲတမ်း ထပ်တလဲလဲ တွေ့ရတဲ့အရာ မဟုတ်တဲ့အတွက် Rule တစ်ခုတည်းနဲ့ Detect လုပ်ဖို့ ပိုပြီး ခက်သွားပါတယ်။

အချို့ အခြေအနေတွေမှာ ETW (Event Tracing for Windows) ကို အသုံးပြုပြီး ပိုမိုအသေးစိတ် Detect လုပ်နိုင်ပေမယ့် အဲ့ဒီလို Capability ကို အဖွဲ့အစည်းတိုင်းမှာ မရှိကြပါဘူး။ အများဆုံး ဖြစ်နိုင်ချေရှိတာက EDR (Endpoint Detection and Response) လို Existing Security Tool တစ်ခုက SharpHound ရဲ့ လှုပ်ရှားမှုကို Suspicious အဖြစ် ဖမ်းမိနိုင်တာ ဖြစ်ပါတယ်။ ဒါမှမဟုတ် Attacker က SharpHound ရဲ့ အခြား Aggressive Feature တွေကို အသုံးပြုတဲ့အခါ ပိုပြီး သိသာလာပြီး Detect ဖြစ်သွားနိုင်ပါတယ်။

SID-History Injection

Active Directory Forest တစ်ခုထဲမှာ Child Domain တစ်ခုကို Attacker က အပြည့်အဝ ထိန်းချုပ်နိုင်သွားတဲ့အခါ Forest တစ်ခုလုံးကို ထိခိုက်နိုင်တဲ့ လမ်းကြောင်းတွေ အများကြီး ရှိလာပါတယ်။ အဲ့ဒီထဲမှာ အများဆုံးတွေ့ရပြီး အန္တရာယ်အရမ်းကြီးတဲ့ နည်းလမ်းတစ်ခုက SID-History Injection ဖြစ်ပါတယ်။ ဒီနည်းလမ်းဟာ Trust Design ကို အခြေခံထားတဲ့ Attack ဖြစ်ပြီး Forest ကို Security Boundary လို့ ခေါ်ရတဲ့ အကြောင်းရင်းကိုလည်း တိတိကျကျ ပြသပေးပါတယ်။

SID History

SID History ဆိုတာ အစပိုင်းမှာ Attack အတွက် မဖန်တီးထားတာပါဘူး။ Domain Migration လုပ်တဲ့အခါ အဆင်ပြေစေဖို့ Microsoft က ထည့်ပေးထားတဲ့ Feature တစ်ခု ဖြစ်ပါတယ်။ User တစ်ယောက် ဒါမှမဟုတ် Group တစ်ခုကို Domain အသစ်တစ်ခုကို ပြောင်းတဲ့အခါ SID အသစ်တစ်ခု ရလာပါတယ်။ SID က Unique ဖြစ်ရတဲ့အတွက် SID ပြောင်းသွားတာနဲ့ User က Domain အဟောင်းမှာ ရထားတဲ့ Permission တွေကို အားလုံး ဆုံးရှုံးသွားနိုင်ပါတယ်။ အဲ့ဒီပြဿနာကို ဖြေရှင်းဖို့ TGT ထဲမှာ Privileged

Attribute Certificate (PAC) ဆိုတဲ့ အပိုင်းထဲကို sidHistory ဆိုတဲ့ Attribute ထည့်ပေးထားပါတယ်။ ဒီ Attribute က User ရဲ့ Domain အဟောင်းမှာ ပါဝင်ခဲ့တဲ့ Group SID တွေကို သိမ်းထားနိုင်အောင် လုပ်ပေးတာပါ။

Impact of SID History

Forest အတွင်းမှာ Authentication လုပ်တဲ့အခါ Domain Controller တွေက sidHistory ထဲက SID တွေကို အပြည့်အဝ ယုံကြည်ပါတယ်။ ဒါကြောင့် sidHistory ထဲမှာ ပါတဲ့ Group SID တွေအတိုင်း Permission တွေကို ချက်ချင်း သက်ရောက်စေပါတယ်။ ဒီအချက်ကြောင့်ပဲ Forest က Active Directory ရဲ့ အမှန်တကယ် Security Boundary ဖြစ်တယ်လို့ ပြောကြတာပါ။ Forest အတွင်းမှာ Privileged Account တစ်ခုရလာရင် အခြား Domain တွေရဲ့ Privilege တွေကိုပါ ထိခိုက်စေနိုင်တဲ့ စွမ်းအား ရှိလာပါတယ်။ အခြား Forest ကို Authentication လုပ်တဲ့အခါတော့ SID Filtering ဆိုတဲ့ Protection ကြောင့် sidHistory ထဲက SID တွေကို ဖယ်ရှားပစ်ပါတယ်။ ဒါက Forest Boundary ကို ကာကွယ်ထားတဲ့ အဓိက Mechanism ဖြစ်ပါတယ်။

Attacking

sidHistory Attribute ကို TGT ဖန်တီးတဲ့အချိန်မှာပဲ သတ်မှတ်ပါတယ်။ ပုံမှန်အခြေအနေမှာ TGT ကို Domain Controller တွေကသာ ဖန်တီးနိုင်ပါတယ်။ ဒါပေမယ့် Attacker က krbtgt Account ရဲ့ Password Hash သို့မဟုတ် Trust Account ရဲ့ Hash ကို ရရှိသွားရင် သူကိုယ်တိုင် TGT ကို ဖန်တီးနိုင်သွားပါတယ်။ Domain Controller က TGT ကို စစ်ဆေးတဲ့အခါ “krbtgt Hash ကို DC ပဲ သိနိုင်တယ်” ဆိုတဲ့ ယုံကြည်ချက်အပေါ် အခြေခံထားပါတယ်။ ဒါပေမယ့် Domain Admin ကို Compromise လုပ်ပြီး Replication Rights ရသွားရင် DCSync Attack ကို သုံးပြီး krbtgt Hash ကို ခိုးယူနိုင်ပါတယ်။

Attacker က ဒီ Hash ကို ရရှိသွားပြီဆိုရင် TGT ရဲ့ အကြောင်းအရာတွေကို လိုသလို ပြင်နိုင်ပါတယ်။ ဒီလို TGT ကို လက်တွေ့မှာ Golden Ticket လို့ ခေါ်ပါတယ်။

> Golden Ticket Attack ဆိုတာ Attacker က User Name၊ Group Membership၊ sidHistory အပါအဝင် PAC ထဲက Attribute တွေအားလုံးကို ကိုယ်တိုင် ထည့်သွင်းဖန်တီးနိုင်တဲ့ Attack ဖြစ်ပါတယ်။

ဒီအခါ sidHistory ထဲကို Forest အဆင့် Privilege ရှိတဲ့ Group SID တွေကို ထည့်လိုက်တာနဲ့ Child Domain User တစ်ယောက်ဟာ Forest တစ်ခုလုံးအတွက် အလွန်မြင့်တဲ့ Permission ရသွားနိုင်ပါတယ်။

SID-History Injection မှာ အများဆုံး ရည်ရွယ်ချက်ထားခံရတဲ့ Group က **Enterprise Administrators Group** ဖြစ်ပါတယ်။

> Enterprise Admin ဆိုတာ Forest အဆင့် Administrator ဖြစ်ပြီး Root Domain ကိုပါ အပြည့်အဝ ထိန်းချုပ်နိုင်ပါတယ်။

Attacker က sidHistory ထဲကို Enterprise Admin Group ရဲ့ SID ကို ထည့်ထားတဲ့ TGT တစ်ခု ဖန်တီးနိုင်ပြီဆိုရင် Root Domain ကို ထိန်းချုပ်နိုင်သွားပြီး Forest တစ်ခုလုံး Compromise ဖြစ်သွားပါပြီ။ Enterprise Admin Group ရဲ့ SID ကို Domain SID နောက်မှာ **RID 519** ကို ထည့်ထားတာနဲ့ ခွဲခြားသိနိုင်ပါတယ်။ ဒီ **RID -519** က Well-Known SID ဖြစ်တဲ့အတွက် Log Analysis လုပ်တဲ့အခါ အရေးကြီးတဲ့ Indicator တစ်ခု ဖြစ်ပါတယ်။

```
external_table('Winlog')
| where EventCode == 1 and CommandLine contains "rubeus.exe" and CommandLine
contains "user:alice"
| project Timestamp, EventCode, HostHostname, CommandLine
```

ပေးထားတဲ့ ဥပမာမှာ Attacker က rubeus.exe ကို အသုံးပြုပြီး alice ဆိုတဲ့ User အတွက် TGT တစ်ခု ဖန်တီးပြီး sidHistory Attribute ထဲကို Enterprise Administrators Group SID ကို ထည့်ထားတာကို တွေ့ရပါတယ်။ ဒီလို Activity တွေကို Log ထဲမှာ တွေ့နိုင်ရင် Forest Compromise အဆင့်ကို ရောက်နေပြီ ဆိုတာကို ခန့်မှန်းနိုင်ပါတယ်။

ဒီအပိုင်းရဲ့ အဓိက သဘောတရားက Child Domain တစ်ခု Compromise ဖြစ်သွားရင် Forest လုံးဝ လုံခြုံနေတယ်လို့ မယုံသင့်ဘူးဆိုတာပါ။ SID-History Injection နဲ့ Golden Ticket Attack တွေကြောင့် Trust Design ကို အသုံးချပြီး Forest တစ်ခုလုံးကို လွယ်လွယ်ကူကူ ထိန်းချုပ်နိုင်ပါတယ်။ ဒါကြောင့် Defender အနေနဲ့ krbtgt Account၊ Trust Account၊ Replication Rights နဲ့ Privileged Account တွေကို အထူး ဂရုစိုက် ကာကွယ်ထားဖို့ အလွန်အရေးကြီးပါတယ်။

Test : Domain Hopping

Which system did the attacker log onto with the golden ticket created in the SID-History Injection section?

အရင်ဆုံး Domain Hopping ဆိုတာကို ပြန်သဘောတရားထားကြည့်ရအောင်။ Domain Hopping ဆိုတာ Attacker တစ်ယောက်က Domain တစ်ခုကို Compromise လုပ်ပြီးနောက် Trust Relationship တွေကို အသုံးချပြီး အခြား Domain တွေကို ဆက်တိုက် ဝင်ရောက်နိုင်သွားတဲ့ အခြေအနေကို ဆိုလိုပါတယ်။ ဒီ Scenario မှာ Attacker က Golden Ticket ကို အသုံးပြုပြီး alice ဆိုတဲ့ User ကို Enterprise-level Privilege ရအောင် ဖန်တီးခဲ့ပါတယ်။ ဒါကြောင့် alice ဟာ သူမူလ Domain ဖြစ်တဲ့ us.aceresponder.local ထဲမှာသာ မကဘဲ Forest အတွင်းရှိ Domain အခြားတွေမှာပါ အခွင့်အရေးမြင့်မားစွာ Authentication လုပ်နိုင်သွားပါတယ်။

ဒီလို Domain Hopping တကယ်ဖြစ်ခဲ့လားဆိုတာကို Defender အနေနဲ့ စစ်ဆေးချင်ရင် Windows Security Event Log ကို အဓိက အသုံးပြုရပါတယ်။ အထူးသဖြင့် Event ID 4624 ကို ကြည့်ရပါတယ်။

> Event 4624 ဆိုတာ Successful Logon ဖြစ်ပြီး User တစ်ယောက်က System တစ်ခုထဲကို အောင်မြင်စွာ Authentication လုပ်နိုင်ခဲ့တယ်ဆိုတာကို ပြပါတယ်။

Golden Ticket ဖန်တီးပြီးနောက် alice က Domain အမျိုးမျိုးမှာ Log on လုပ်နိုင်သွားမယ်ဆိုရင် ဒီ Event တွေ ဆက်တိုက် ပေါ်လာပါလိမ့်မယ်။

```
external_table('Winlog')
| where EventCode == 4624 and TargetUserName contains "alice" and
TargetDomainName contains "us.aceresponder.local" and IpAddress ==
"10.0.0.10" and Timestamp > datetime("2023-10-20T16:25:45.837Z")
| project Timestamp, EventCode, EventAction, Message, HostHostname
```

ပေးထားတဲ့ Query မှာ alice-pc ကနေ လာတဲ့ 4624 Event တွေကို Golden Ticket ဖန်တီးခဲ့တဲ့ အချိန်နောက်ပိုင်းမှာပဲ စစ်ထားပါတယ်။ ဒီလိုလုပ်တာက Golden Ticket မရှိခင် Normal Authentication တွေနဲ့ ခွဲခြားဖို့ ဖြစ်ပါတယ်။ TargetUserName ထဲမှာ alice ပါတဲ့ Event တွေကိုသာ ရွေးထားပြီး TargetDomainName ကို us.aceresponder.local လို့ ကန့်သတ်ထားတာက alice ဟာ သူ့ Domain ထဲမှာ Enterprise-level Privilege နဲ့ Authentication လုပ်နေတာကို သက်သေပြချင်လို့ ဖြစ်ပါတယ်။ IP Address ကိုပါ ကန့်သတ်ထားတာက Attacker ရဲ့ Machine ကနေ ဖြစ်လာတဲ့ Activity ကို သေချာစွာ ခြေရာခံနိုင်အောင် ဖြစ်ပါတယ်။

```
An account was successfully logged on.

Subject:
        Security ID:            S-1-0-0
        Account Name:           -
        Account Domain:         -
        Logon ID:               0x0

Logon Information:
        Logon Type:             3
        Restricted Admin Mode:  -
        Virtual Account:                No
        Elevated Token:         Yes

Impersonation Level:            Delegation

New Logon:
        Security ID:            S-1-5-21-3954728291-3628540421-2748508761-1603
        Account Name:           alice
        Account Domain:         US.ACERESPONDER.LOCAL
        Logon ID:               0x511ACE
        Linked Logon ID:                0x0
        Network Account Name:   -
        Network Account Domain: -
        Logon GUID:             {c4c9e287-8dde-9e53-4c1c-894071dfc88e}

Process Information:
        Process ID:             0x0
        Process Name:           -

Network Information:
        Workstation Name:       -
        Source Network Address: 10.0.0.10
        Source Port:            56750

Detailed Authentication Information:
        Logon Process:          Kerberos
        Authentication Package: Kerberos
        Transited Services:     -
        Package Name (NTLM only):       -
        Key Length:             0
```

ဒီနေရာမှာ အရေးကြီးတဲ့ အသေးစိတ်တစ်ခုက .keyword field ကို အသုံးပြုထားတာပါ။ Windows Log တွေကို SIEM ထဲ ထည့်တဲ့အခါ Text Field တွေကို Tokenize လုပ်ထားတတ်ပါတယ်။ အဲ့ဒီလို Tokenize ဖြစ်ထားရင် alice-pc$ လို Computer Account Authentication တွေပါ Query ထဲ ပါဝင်လာနိုင်ပါတယ်။ Defender အနေနဲ့ User Account ဖြစ်တဲ့ alice ကိုပဲ တိတိကျကျ ရွေးချင်တဲ့အတွက် TargetUserName.keyword ကို အသုံးပြုပြီး Full Text Match လုပ်ထားတာပါ။ ဒီလိုလုပ်မှ alice-pc$ လို Machine Account Authentication တွေကို Filter လုပ်ပစ်နိုင်ပါတယ်။

Test : Golden Ticket

Which share did the attacker access on dc with the golden ticket?

> Golden Ticket Attack ဆိုတာ Attacker က krbtgt Account ရဲ့ Password Hash ကို ရရှိပြီး ကိုယ်တိုင် TGT တစ်ခုကို ဖန်တီးတဲ့ Attack ဖြစ်ပါတယ်။

ဒီ TGT က တရားဝင် Domain Controller က ထုတ်ပေးထားသလိုပဲ ဖြစ်နေတာကြောင့် Domain Controller ကလည်း ယုံကြည်ပြီး Authentication ကို အောင်မြင်အောင် ခွင့်ပြုပေးပါတယ်။ ဒါပေမယ့် Defender အနေနဲ့ “Ticket ဖန်တီးထားတယ်” ဆိုတာတစ်ခုတည်းနဲ့ မလုံလောက်ပါဘူး။ အဲဒီ Ticket ကို အသုံးပြုပြီး Resource တွေကို တကယ် Access လုပ်နိုင်နေပြီလားဆိုတာကို Log တွေနဲ့ သက်သေပြရပါမယ်။

ဒီမှာ အဓိကကြည့်ရတဲ့ Event ID က 5140 ဖြစ်ပါတယ်။

> Event 5140 ဆိုတာ Network Share Object ကို Access လုပ်တဲ့အခါ ပေါ်လာတဲ့ Event ဖြစ်ပြီး Domain Environment မှာ User တစ်ယောက်က Server တစ်ခုရဲ့ Share ကို Access လုပ်နိုင်ခဲ့တယ် ဆိုတဲ့ အဓိပ္ပါယ်ကို ပြပါတယ်။

Domain Controller တစ်ခုကို Access လုပ်တဲ့အခါ SYSVOL သို့မဟုတ် NETLOGON လို Default Share တွေကို မဖြစ်မနေ Access လုပ်ရတာကြောင့် Golden Ticket အောင်မြင်ရင် ဒီ Event ကို ချက်ချင်း တွေ့ရတတ်ပါတယ်။

```
external_table('Winlog')
| where Timestamp between (datetime("2023-10-20T16:25:47.736Z") ..
datetime("2023-10-20T16:25:48.736Z")) and EventCode == 5140 and
SubjectUserName contains "alice"
| project Timestamp, EventCode, EventAction, Message
```

ပေးထားတဲ့ Query မှာ dc.aceresponder.local ကို Authentication လုပ်ခဲ့တဲ့ အချိန်အနားမှာပဲ Event 5140 ကို ရှာထားပါတယ်။ Time Window ကို အလွန်တိုတောင်းစွာ သတ်မှတ်ထားတာက Authentication နဲ့ Share Access ကို တစ်ဆက်တည်း ဖြစ်ပေါ်လာတဲ့ Activity အဖြစ် ချိတ်ဆက်ကြည့်ချင်လို့ ဖြစ်ပါတယ်။ EventCode ကို 5140 လို့ ကန့်သတ်ထားပြီး SubjectUserName ထဲမှာ alice ပါတဲ့ Event တွေကို ရွေးထားတာက Golden Ticket နဲ့ ဖန်တီးထားတဲ့ alice Account ကို အသုံးပြုပြီး Access ဖြစ်နေတာကို သက်သေပြချင်လို့ ဖြစ်ပါတယ်။

ဒီလို Event 5140 ကို Authentication ပြီးချင်း တွေ့ရတယ်ဆိုရင် alice ဟာ Domain Controller ရဲ့ Network Share တွေကို Access လုပ်နိုင်နေပြီဆိုတာကို ဆိုလိုပါတယ်။ ပုံမှန် User တစ်ယောက်အတွက် Domain Controller ရဲ့ Share တွေကို Access လုပ်နိုင်ဖို့ မလွယ်ကူပါဘူး။ ဒါကြောင့် ဒီ Event တွေဟာ Golden Ticket Attack အောင်မြင်ပြီး Privilege မြင့်မားတဲ့ Access ကို ရရှိထားပြီဆိုတဲ့ အရေးကြီးတဲ့ Indicator တစ်ခု ဖြစ်ပါတယ်။

SID-History Injection 2

The attacker created a second golden ticket using the krbtgt hash. Which account did the attacker use for the ticket?

အရင် Golden Ticket Recap မှာ alice အတွက် Enterprise-level Privilege ပါတဲ့ Golden Ticket တစ်ခုကို ဖန်တီးပြီး Domain Hopping လုပ်နိုင်ခဲ့တာကို တွေ့ခဲ့ပါတယ်။ ဒါပေမယ့် Attacker တစ်ယောက်က အဲ့ဒီအဆင့်ရောက်သွားပြီဆိုရင် တစ်ခုတည်းနဲ့ မရပ်တန့်တတ်ပါဘူး။ အများအားဖြင့် Privilege အမျိုးမျိုး၊ Identity အမျိုးမျိုးနဲ့ Golden Ticket တွေကို ထပ်မံ ဖန်တီးပြီး Persistence နဲ့ Redundancy ကို တည်ဆောက်တတ်ပါတယ်။ ဒီ Recap က အဲ့ဒီလို "ဒုတိယ Golden Ticket" ကို ဘယ်လို ဖော်ထုတ်မလဲဆိုတာကို ပြထားတာ ဖြစ်ပါတယ်။

ဒီအဆင့်မှာ Defender အနေနဲ့ ရှာဖွေသုံးတဲ့ နည်းလမ်းက အလွန်ရိုးရှင်းပါတယ်။ Windows Event Log ထဲမှာ Process Creation Event ဖြစ်တဲ့ Event ID 1 ကို ကြည့်ပြီး rubeus.exe ကို အသုံးပြုထားတဲ့ Command Line တွေကို ရှာပါတယ်။ အထူးသဖြင့် CommandLine ထဲမှာ "golden" ဆိုတဲ့ စကားလုံး ပါနေတယ်ဆိုရင် Golden Ticket ဖန်တီးနေတဲ့ Activity ဖြစ်နိုင်ချေ အလွန်မြင့်ပါတယ်။ Rubeus မှာ Golden Ticket ဖန်တီးတဲ့ Command တွေမှာ golden ဆိုတဲ့ Keyword ကို မဖြစ်မနေ အသုံးပြုရတာကြောင့် ဒီဟာက အရေးကြီးတဲ့ Detection Point ဖြစ်ပါတယ်။

```
external_table('Winlog')
| where EventCode == 1 and CommandLine contains "rubeus.exe" and CommandLine
contains "golden"
| project EventCode, HostHostname, EventAction, CommandLine
```

ဒီ Query ကို Run လိုက်တဲ့အခါ Golden Ticket ဖန်တီးတဲ့ Command တစ်ခုထက် ပိုပြီး တွေ့ရပါတယ်။ အဲ့ဒီထဲမှာ alice အတွက် ဖန်တီးထားတဲ့ Golden Ticket အပြင် RID 500 ပါတဲ့ Administrator Account ကို အသုံးပြုထားတဲ့ Golden Ticket Command တစ်ခုလည်း ပါဝင်နေတာကို တွေ့ရပါတယ်။ RID 500 ဆိုတာ Built-in Administrator Account ရဲ့ Well-Known RID ဖြစ်ပါတယ်။ Domain တိုင်းမှာ Administrator Account ရဲ့ SID ဟာ Domain SID နောက်မှာ -500 နဲ့ အဆုံးသတ်ပါတယ်။ ဒါကြောင့် Log ထဲမှာ RID 500 ပါတဲ့ Golden Ticket ကို တွေ့ရင် Attacker က Forest သို့မဟုတ် Domain အဆင့် Built-in Administrator Identity ကို တိုက်ရိုက် အတုလုပ်ထားတယ်ဆိုတာကို ချက်ချင်း သဘောပေါက်နိုင်ပါတယ်။

ဒီအချက်က ဘာကို ပြသလဲဆိုရင် Attacker ဟာ alice Identity တစ်ခုတည်းကို မသုံးတော့ဘဲ Administrator Account ကိုပါ အတုလုပ်ပြီး Golden Ticket ဖန်တီးထားတယ်ဆိုတာပါ။ ဒါဟာ Defender အနေနဲ့ အလွန်အန္တရာယ်ကြီးတဲ့ အခြေအနေတစ်ခုဖြစ်ပါတယ်။ User Account တစ်ခု Compromise ဖြစ်နေတာထက် Built-in Administrator Identity ကို Golden Ticket နဲ့ ဖန်တီးထားတာက Detection ပိုခက်သလို Cleanup လုပ်ရတာလည်း ပိုပြီး ခက်ခဲပါတယ်။

SID-History Adendum

Golden Ticket ကို အသုံးပြုပြီး User တစ်ယောက်က အောင်မြင်စွာ Log on ဝင်နိုင်ခဲ့ပြီဆိုရင် Windows က **Logon Success Event ဖြစ်တဲ့ 4624** ကို အရင်ဆုံး မှတ်တမ်းတင်ပါတယ်။ ဒါပေမယ့် **User ဟာ ဘယ် Group တွေရဲ့ အခွင့်အရေးတွေကို တကယ် ရယူထားလဲဆိုတာကို 4624 Event တစ်ခုတည်းနဲ့ မသိနိုင်ပါဘူး။** ဒီနေရာမှာ အရေးကြီးတဲ့ Event တစ်ခုက 4627 ဖြစ်ပါတယ်။

> Event 4627 ကို "Group membership information" Event လို့ ခေါ်ပြီး Successful Logon ဖြစ်ပြီးချင်းမှာ ထပ်မံ ထုတ်ပေးပါတယ်။

Event 4627 မှာ User တစ်ယောက် Log on ဝင်တဲ့အချိန် သူ့အတွက် အမှန်တကယ် သက်ရောက်နေတဲ့ Group Membership တွေကို မှတ်တမ်းတင်ထားပါတယ်။ အဲဒီ Group Membership တွေက User ရဲ့ မူလ Group Membership တွေချည်းမကဘဲ Kerberos Ticket ထဲမှာ ပါလာတဲ့ Effective Group Membership တွေ အားလုံးကို ပါဝင်ပါတယ်။ ဒါကြောင့် Golden Ticket နဲ့ sidHistory Injection လုပ်ထားတဲ့အခါ အဲဒီ Injection လုပ်ထားတဲ့ Group SID တွေကိုပါ ဒီ Event ထဲမှာ မြင်နိုင်ပါတယ်။

```
Group membership information.

Subject:
        Security ID:            S-1-0-0
        Account Name:           -
        Account Domain:         -
        Logon ID:               0x0

Logon Type:                     3

New Logon:
        Security ID:            S-1-5-21-3954728291-3628540421-2748508761-1603
        Account Name:           alice
        Account Domain:         US.ACERESPONDER.LOCAL
        Logon ID:               0x511ACE

Event in sequence:              1 of 1

Group Membership:
                %{S-1-5-21-3954728291-3628540421-2748508761-513}
                %{S-1-1-0}
                %{S-1-5-32-554}
                %{S-1-5-32-545}
                %{S-1-5-32-544}
                %{S-1-5-2}
                %{S-1-5-11}
                %{S-1-5-15}
                %{S-1-5-21-2664032064-894357479-207739468-519}
                %{S-1-5-21-2664032064-894357479-207739468-572}
                %{S-1-16-12288}
```

အထူးသဖြင့် Attacker က Golden Ticket ထဲမှာ Enterprise-level Group တစ်ခုကို sidHistory ထဲ Inject လုပ်ထားတယ်ဆိုရင် Event 4627 ရဲ့ GroupMembership Field ထဲမှာ အဲဒီ Group ရဲ့ SID ကို တိတိကျကျ တွေ့ရပါမယ်။ ဥပမာ Enterprise Administrators Group ရဲ့ SID ဖြစ်တဲ့ Domain SID နောက်မှာ -519 ပါတဲ့ SID ကို တွေ့ရရင် ဒီ User ဟာ Enterprise Admin Privilege ကို Golden Ticket နဲ့ အတုလုပ်ထားတယ်ဆိုတာကို ချက်ချင်း ခန့်မှန်းနိုင်ပါတယ်။

ဒီ Event ရဲ့ အရေးကြီးချက်က Defender အနေနဲ့ Attack ကို Scope လုပ်တဲ့အခါ အလွန်အသုံးဝင်တာပါ။ Golden Ticket ဖန်တီးထားတယ်ဆိုတာကို သံသယရှိရုံနဲ့ မရပါဘူး။ ဘယ် User တွေက Enterprise-level Privilege ရနေပြီလဲ၊ ဘယ် System တွေမှာ အဲဒီ Ticket ကို အသုံးပြုပြီး Log on လုပ်ပြီးပြီလဲ ဆိုတာကို မြန်မြန်ဆန်ဆန် သိရပါမယ်။ Event 4627 ကို အသုံးပြုရင် Log on ဖြစ်ခဲ့တဲ့ User ရဲ့ Effective Privilege ကို တိုက်ရိုက် မြင်နိုင်တာကြောင့် Investigation ကို အလွန်မြန်စေပါတယ်။

GoldenGMSA

Group Managed Service Account (gMSA) ဆိုတာ ပုံမှန် Service Account တစ်မျိုးပါပဲ။ Service Account ဆိုတာ Application တွေ၊ Service တွေကို Run ဖို့ အသုံးပြုတဲ့ Account ဖြစ်ပါတယ်။ ပုံမှန် Service Account တွေရဲ့ပြဿနာက Password ကို လူက ကိုယ်တိုင် သတ်မှတ်ရပြီး မပြောင်းဘဲ အချိန်ကြာကြာ အသုံးပြုနေတတ်တာပါ။ ဒါကြောင့် Password က အားနည်းတတ်ပြီး Crack လုပ်လို့ လွယ်တတ်ပါတယ်။ ဒီပြဿနာကို ဖြေရှင်းဖို့ Microsoft က gMSA ကို မိတ်ဆက်ခဲ့တာဖြစ်ပြီး Password ကို Domain Controller က အလိုအလျောက် ပုံမှန်ပြောင်းပေးပါတယ်။ လူက Password ကို သိစရာ မလိုတော့တဲ့အတွက် Security ပိုကောင်းလာပါတယ်။

ဒါပေမယ့် ဒီ Automatic Password Rotation ကို လုပ်ဖို့အတွက် အခြေခံ Secret တစ်ခု လိုအပ်ပါတယ်။ အဲဒီ Secret ကို **KDS root key** လို့ ခေါ်ပါတယ်။ KDS root key ကို Active Directory ထဲမှာ **msKds-ProvRootKey** ဆိုတဲ့ Object အနေနဲ့ သိမ်းထားပါတယ်။ Windows က ဒီ Root Key ကို gMSA Object ထဲက အခြား Attribute တွေနဲ့ ပေါင်းပြီး gMSA Password ကို တွက်ချက်ပါတယ်။ အဓိက အချက်က Password ကို တိုက်ရိုက် သိမ်းထားတာ မဟုတ်ဘဲ Root Key ကို အသုံးပြုပြီး တွက်ချက်နေတာပါ။

Risk

ဒီနေရာမှာ အန္တရာယ်ကြီးတဲ့ သဘောတရားတစ်ခု ပေါ်လာပါတယ်။ Active Directory Forest တစ်ခုမှာ **Configuration Context** ဆိုတဲ့ အပိုင်းကို Forest အတွင်းရှိ Domain အားလုံးကို Replicate လုပ်ပါတယ်။ msKds-ProvRootKey Object က Configuration Context ထဲမှာ ရှိတဲ့အတွက် KDS root key က Child Domain အားလုံးကိုပါ Replicate ဖြစ်သွားပါတယ်။ အဲ့ဒီအတွက် Forest ထဲက Domain Controller တစ်လုံးမဆို System-level Access ရသွားရင် KDS root key ကို ခိုးယူနိုင်တဲ့ အခွင့်အရေး ရလာပါတယ်။

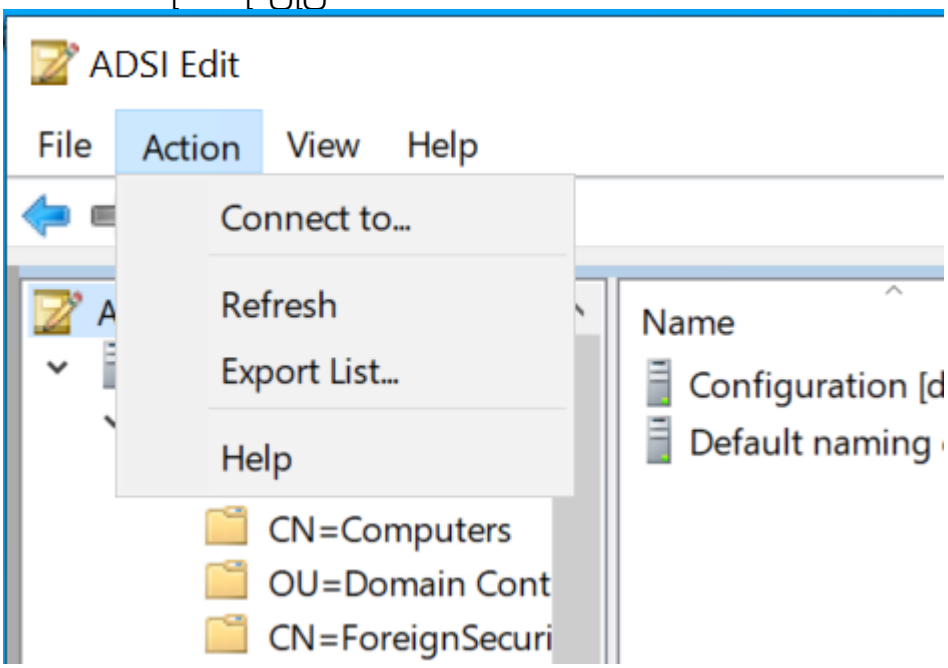
Attack

Attacker က Child Domain ရဲ့ Domain Controller တစ်လုံးကို SYSTEM Level နဲ့ Compromise လုပ်နိုင်သွားပြီဆိုရင် KDS root key ကို ဖတ်နိုင်ပါတယ်။ ဒါနဲ့တင် မပြီးသေးပါဘူး။ gMSA Object ထဲမှာ Password တွက်ချက်ဖို့လိုတဲ့ Attribute တွေကိုပါ ရသွားရင် Attacker က gMSA Password ကို Offline မှာ ကိုယ်တိုင် ပြန်တွက်နိုင်ပါတယ်။ ဒီလိုနဲ့ Root Domain မှာ အသုံးပြုနေတဲ့ gMSA Account ရဲ့ Password ကိုပါ ရရှိသွားနိုင်ပြီး Domain Hopping လုပ်နိုင်သွားပါတယ်။ ဒါကြောင့် ဒီ Attack ကို **GoldenGMSA** လို့ ခေါ်ကြတာပါ။

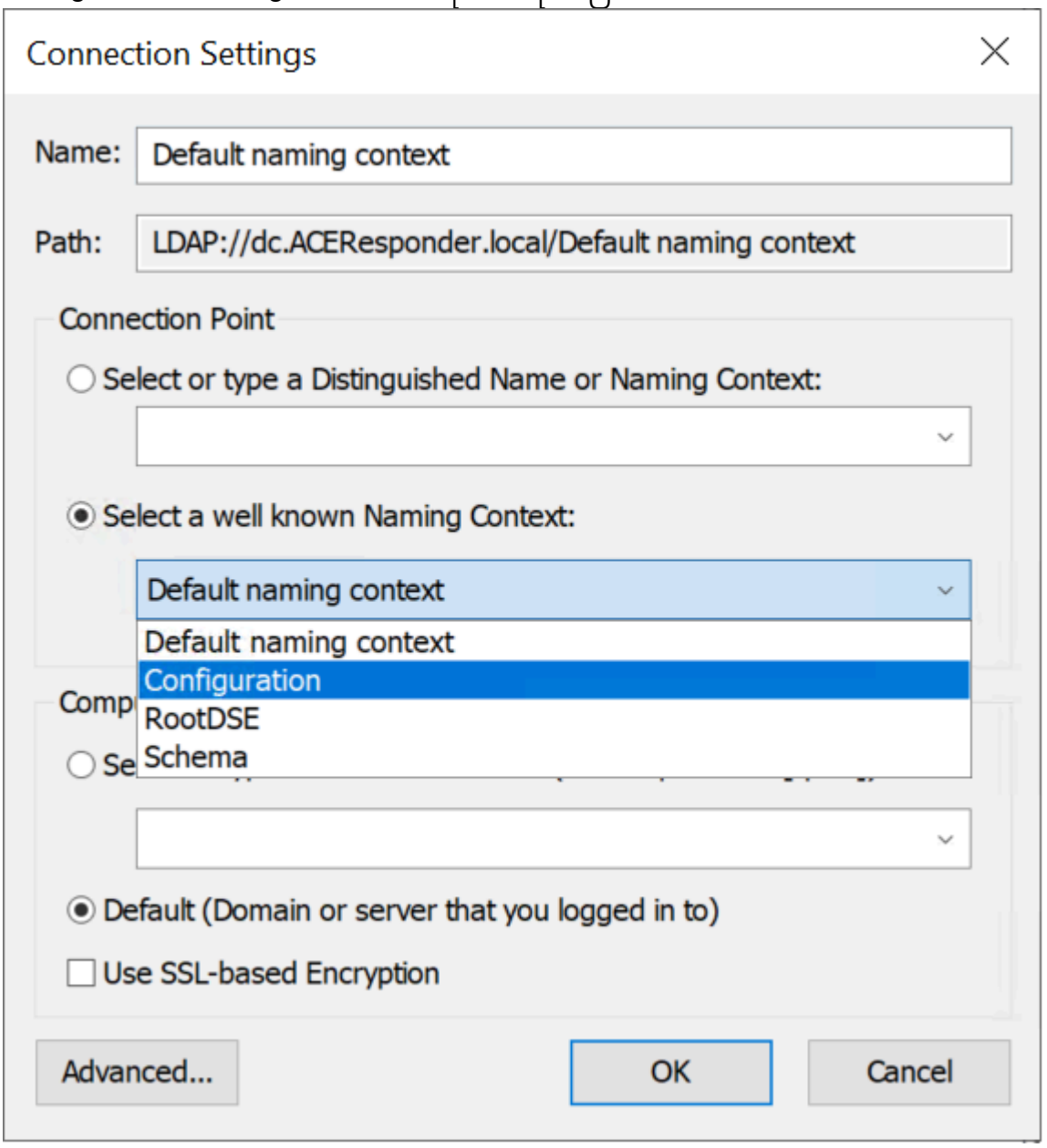
Detection

ဒီအဆင့်မှာ Defender အနေနဲ့ အရေးကြီးဆုံး မေးခွန်းက “Attacker က KDS root key ကို ဖတ်နေတယ်ဆိုတာကို ဘယ်လို သိနိုင်မလဲ” ဆိုတာပါ။ ပုံမှန်အားဖြင့် msKds-ProvRootKey Object ကို ဖတ်တာကို Log မချထားပါဘူး။ ဒါကြောင့် Detection လုပ်ချင်ရင် Audit Setting ကို ကိုယ်တိုင် Configure လုပ်ရပါမယ်။ Audit Object Access ကို Enable လုပ်ပြီး msKds-ProvRootKey Object အပေါ် SACL ကို သတ်မှတ်ရပါမယ်။

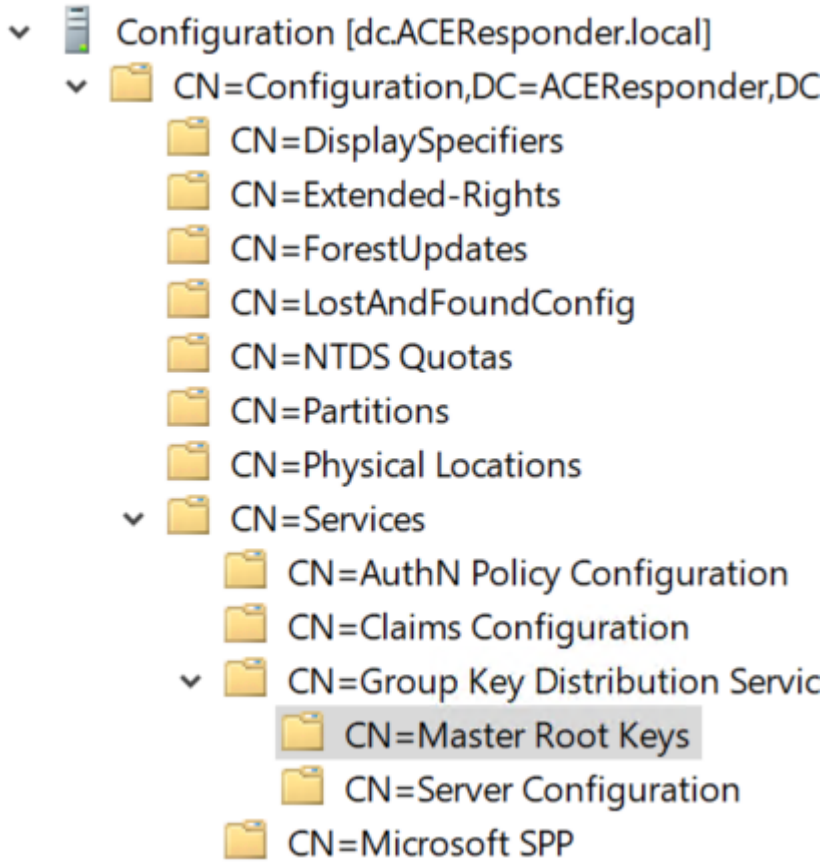
1. ADSI Edit ကို အသုံးပြုပြီး



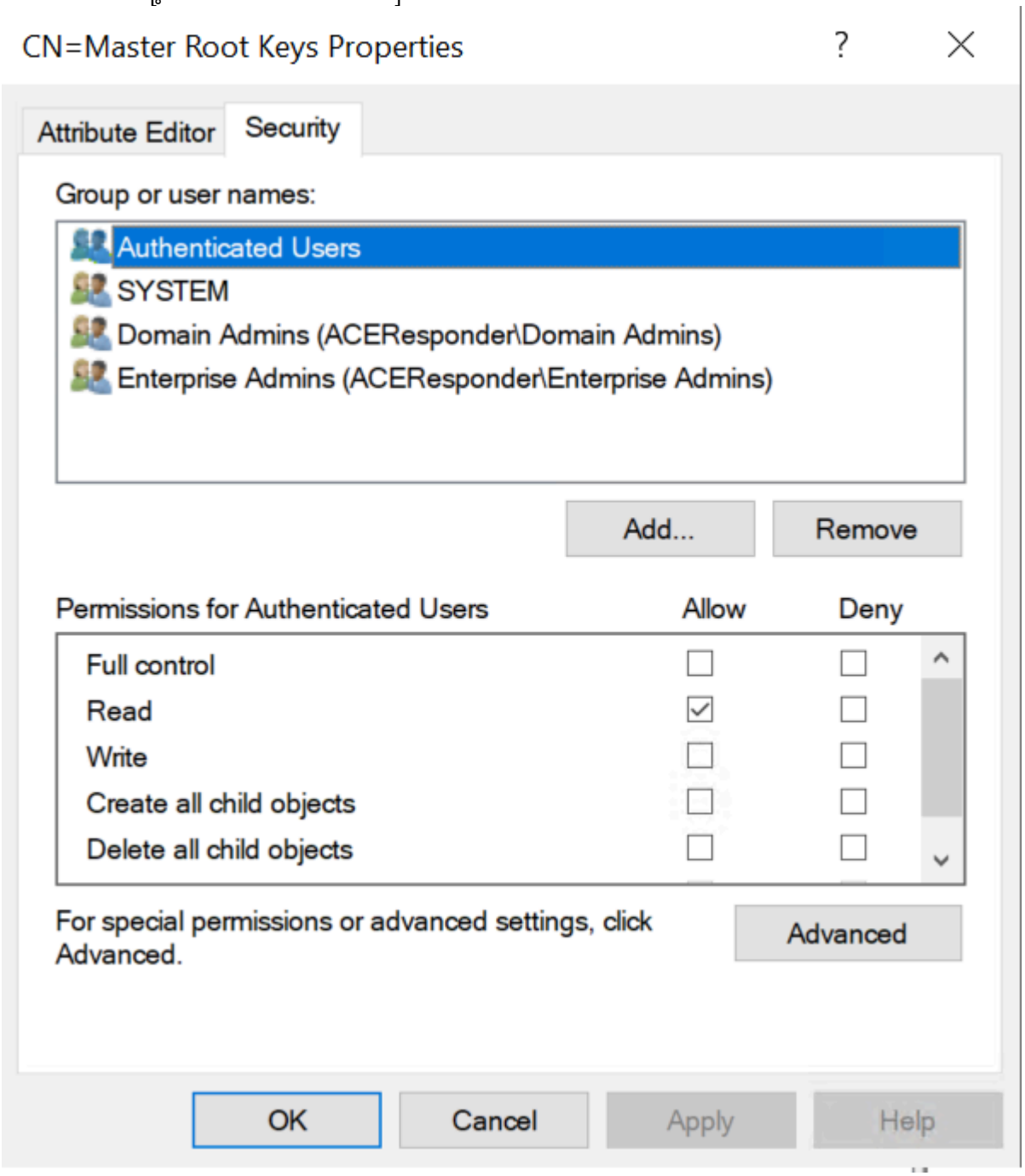
2. Configuration Naming Context ထဲကို ဝင်ရောက်ပြီး



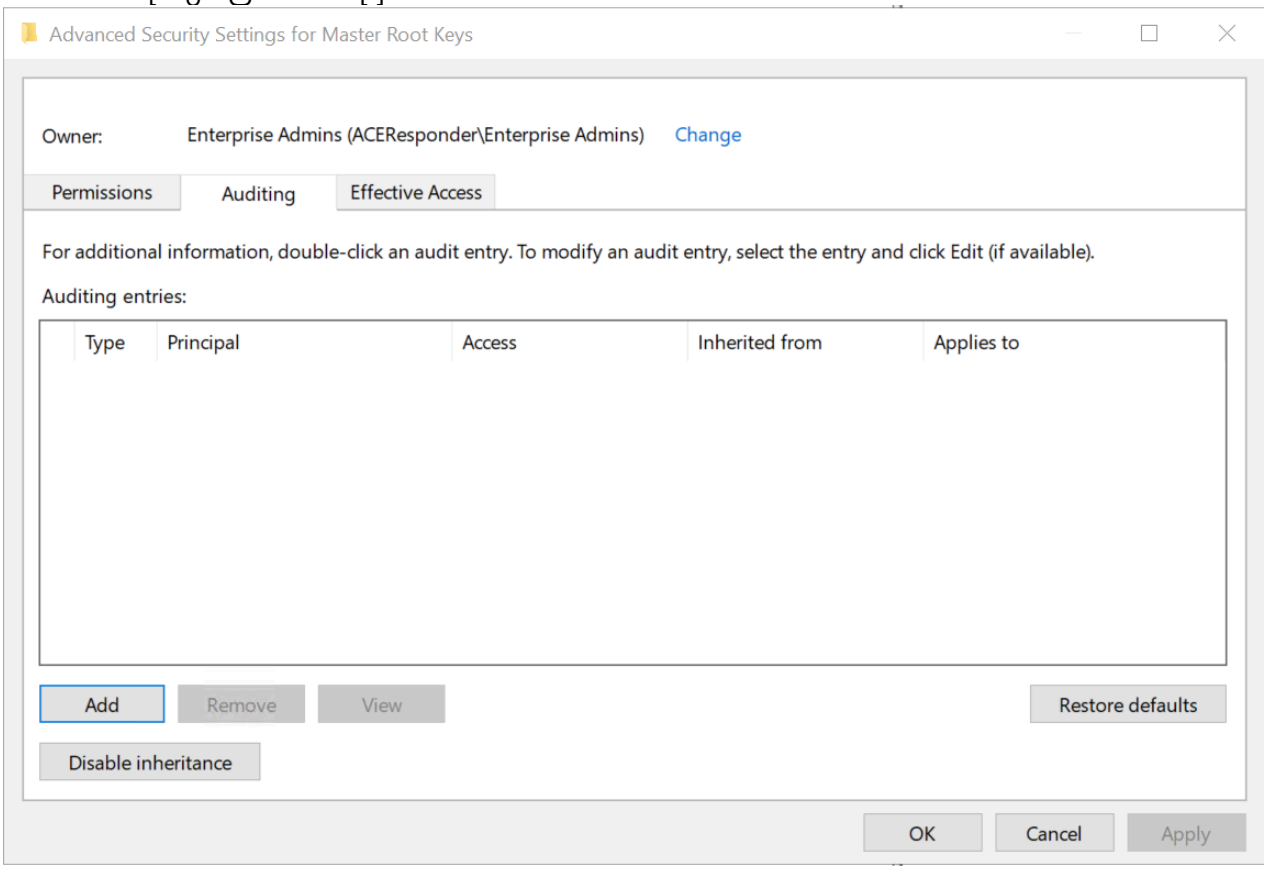
3. Group Key Distribution Service အောက်မှာရှိတဲ့ Master Root Keys ကို ရွေးပါတယ်။



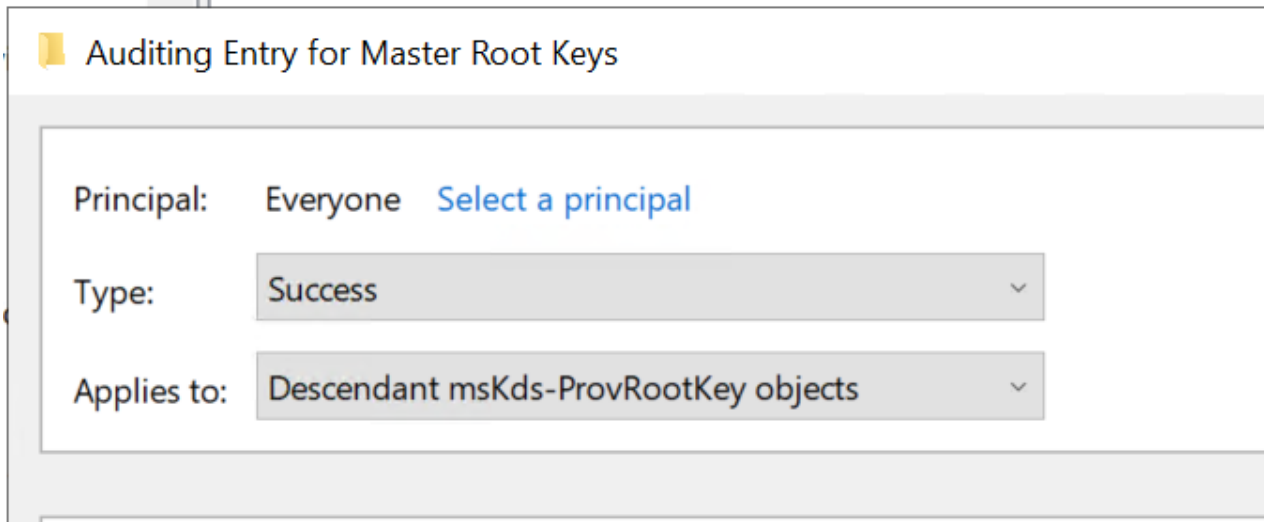
4. အဲဒီ Object ရဲ့ Security Setting ထဲမှာ



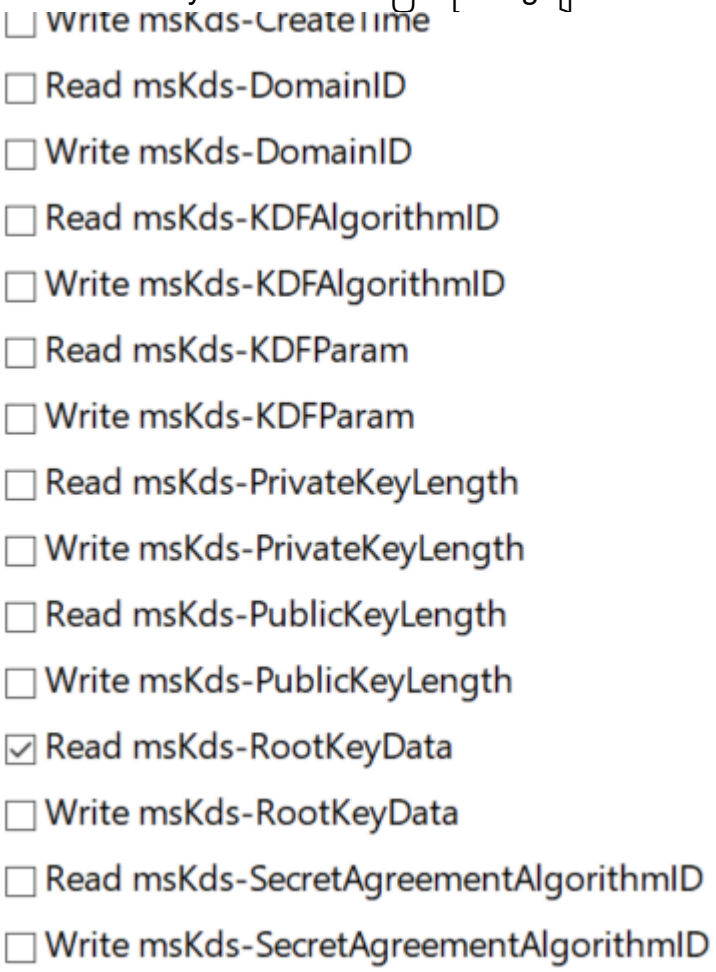
5. Auditing ကို သွားပြီး Add ကိုနှိပ်



6. Everyone Principal အတွက် Descendant msKds-ProvRootKey Object တွေကို ရွေး



7. Read msKds-RootKeyData Access ဖြစ်ရင် Log ချအောင် သတ်မှတ်ပါတယ်။



8. ဒီလို ပြင်ဆင်ပြီးသွားရင် KDS root key ကို ဘယ်သူမဆို ဖတ်တဲ့အချိန်တိုင်း Event Log ထဲမှာ မှတ်တမ်းတင်လာပါလိမ့်မယ်။

```
external_table('Winlog')
| where EventCode == 4662 and ObjectName contains "a6cb1869-d545-49d1-b829-10c6037ebe7c"
| project Timestamp, EventCode, Message
```

ဒီ Audit ကို Enable လုပ်ပြီးနောက် Event ID 4662 ကို ကြည့်ရင် Attacker Activity ကို ဖမ်းမိနိုင်ပါတယ်။

> Event 4662 ဆိုတာ Directory Object Access Event ဖြစ်ပြီး Object Attribute တစ်ခုကို ဖတ်တဲ့အခါ ပေါ်လာပါတယ်။

Log ထဲမှာ msKds-RootKeyData Attribute ကို Access လုပ်ထားတယ်ဆိုရင် KDS root key ကို ဖတ်ထားတာ ဖြစ်ပါတယ်။

```
An operation was performed on an object.

Subject :
        Security ID:            S-1-5-18
        Account Name:           US-DC$
        Account Domain:         US
        Logon ID:               0x581EBA

Object:
        Object Server:          DS
        Object Type:            %{aa02fd41-17e0-4f18-8687-b2239649736b}
        Object Name:            %{a6cb1869-d545-49d1-b829-10c6037ebe7c}
        Handle ID:              0x0

Operation:
        Operation Type:         Object Access
        Accesses:               Read Property

        Access Mask:            0x10
        Properties:             Read Property
                {771727b1-31b8-4cdf-ae62-4fe39fadf89e}
                        {26627c27-08a2-0a40-a1b1-8dce85b42993}
        {aa02fd41-17e0-4f18-8687-b2239649736b}


Additional Information:
        Parameter 1:            -
        Parameter 2:
```

ပေးထားတဲ့ ဥပမာမှာ US-DC ကနေ msKds-RootKeyData ကို Access လုပ်ထားတာကို တွေ့ရပြီး Child Domain Domain Controller တစ်လုံးက Forest-wide Secret ကို ဖတ်နိုင်နေပြီဆိုတဲ့ အန္တရာယ်ကြီးတဲ့ အခြေအနေကို ပြသနေပါတယ်။

GoldenGMSA Account

What is the name of the executable the attacker used to compromise the KDS Root Key?

အရင်ဆုံး သတိထားရမယ့် အချက်က Event ID 4662 ပါ။ Event 4662 ပေါ်လာတယ်ဆိုတာ Directory Object တစ်ခုရဲ့ Sensitive Attribute ကို Access လုပ်ထားတယ်ဆိုတဲ့ အဓိပ္ပါယ်ပါ။ ဒီ Scenario မှာတော့ msKds-ProvRootKey Object ထဲက msKds-RootKeyData ကို ဖတ်ထားတာဖြစ်ပြီး KDS root key ကို ခိုးယူနေတဲ့ လက္ခဏာဖြစ်ပါတယ်။ ဒါပေမယ့် 4662 Event တစ်ခုတည်းကြည့်ရုံနဲ့ "ဘာ Tool နဲ့ ဖတ်သလဲ" ဆိုတာကို မသိနိုင်သေးပါဘူး။

ဒါကြောင့် Defender အနေနဲ့ လုပ်ရမယ့်အရာက 4662 Event မပေါ်လာခင် အချိန်အနည်းငယ်အတွင်းမှာ ဘာ Process တွေ Run ဖြစ်ခဲ့လဲဆိုတာကို ပြန်ကြည့်တာပါ။ အဲဒီအတွက် Event ID 1 ကို အသုံးပြုပါ

တယ်။ Event ID 1 က Process Creation Event ဖြစ်ပြီး ဘယ် Program ကို ဘယ် Command Line နဲ့ Run ခဲ့လဲဆိုတာကို အသေးစိတ် ပြပေးပါတယ်။

```
external_table('Winlog')
| where EventCode == 1 and HostHostname contains "us-dc" and Timestamp
between (datetime("2023-10-20T17:00:45.302Z") .. datetime("2023-10-
20T17:00:47.302Z"))
| project Timestamp, EventCode, HostHostname, User, CommandLine
```

ပေးထားတဲ့ Query က US Domain Controller ဖြစ်တဲ့ us-dc ပေါ်မှာ 4662 Event မပေါ်ခင် စက္ကန့် အနည်းငယ်အတွင်း Process Creation ဖြစ်ခဲ့တာတွေကို ရှာဖွေထားတာပါ။ အဲဒီ Query ရဲ့ Result ကို ကြည့်လိုက်ရင် `"C:\users\administrator\desktop\GoldenGMSA.exe" kdsinfo -f us.aceresponder.local` ဆိုတဲ့ Command Line နဲ့ Process တစ်ခု Run ဖြစ်နေတာကို တွေ့ရပါတယ်။

ဒီအချက်က အရမ်းအရေးကြီးပါတယ်။ GoldenGMSA.exe ဆိုတဲ့ Tool ကို Administrator Account နဲ့ Domain Controller ပေါ်မှာ တိုက်ရိုက် Run ထားပြီး `kdsinfo` ဆိုတဲ့ Option ကို အသုံးပြုပြီး Forest Domain ဖြစ်တဲ့ us.aceresponder.local ရဲ့ KDS Information ကို ဖတ်ထားတာဖြစ်ပါတယ်။ ဒါဟာ တိုက်ရိုက်ပဲ KDS root key ကို Access လုပ်ဖို့ ကြိုးစားနေတဲ့ လုပ်ရပ်ဖြစ်ပြီး အနောက်မှာ ဖြစ်ပေါ်လာတဲ့ 4662 Event နဲ့ တိတိကျကျ ကိုက်ညီပါတယ်။

Inter-Trust Kerberoasting

အဖွဲ့အစည်းတစ်ခုက တခြား ကုမ္ပဏီတစ်ခုကို ဝယ်ယူလိုက်တဲ့အခါ၊ ဒါမှမဟုတ် အပြင်အဖွဲ့အစည်းနဲ့ လက်တွဲအလုပ်လုပ်ရတဲ့အခါ Domain တစ်ခုနဲ့ တစ်ခု ချိတ်ဆက်ဖို့ **Forest Trust** ကို သုံးလေ့ရှိပါတယ်။ အများအားဖြင့် “အပြင်က Network က မလုံခြုံလောက်ဘူး၊ Forest ကွဲထားတာပဲဆိုတော့ အန္တရာယ်နည်းမယ်” လို့ ထင်တတ်ကြပါတယ်။ ဒါပေမယ့် ဒီစာပိုဒ်က အဲဒီအတွေးအခေါ်ကို မမှန်ကြောင်း ပြသထားပါတယ်။ Forest Trust တစ်ခုရှိပြီဆိုရင် အန္တရာယ်က တကယ်ရှိနေပါသေးတယ်။

Trust Account

Trust Authentication အပိုင်းမှာ ပြောခဲ့သလို Domain နှစ်ခုကြား Trust တည်ဆောက်တဲ့အခါ Domain တိုင်းမှာ **Trust Account** ဆိုတဲ့ အထူး Account တစ်ခုစီ အလိုအလျောက် ဖန်တီးပေးပါတယ်။ ဥပမာ အနေနဲ့ ဒီ Scenario မှာ Domain နှစ်ခုရှိပါတယ်။ တစ်ခုက **ACEResponder Forest** ဖြစ်ပြီး တစ်ခုက **Acquisition Forest** ဖြစ်ပါတယ်။ ACEResponder Domain ထဲမှာ **ACQUISITION$** ဆိုတဲ့ Trust Account တစ်ခု ရှိပြီး Acquisition Domain ထဲမှာ **ACERESPONDER$** ဆိုတဲ့ Trust Account တစ်ခု ရှိပါတယ်။

Risk

ဒီ Trust Account တွေရဲ့အရေးကြီးတဲ့အချက်က Domain Admin မဟုတ်ဘဲ **Domain Users group** ထဲမှာ ပါဝင်နေတယ်ဆိုတာပါ။ ဒါကြောင့် “အထူး Account” လို့ ခေါ်ပေမယ့် လက်တွေ့မှာ Domain User တစ်ယောက်လိုပဲ Authentication လုပ်လို့ရပြီး Service Ticket တွေ တောင်းလို့ရပါတယ်။ အကယ်၍

Attacker တစ်ယောက်က ဒီ Trust Account ရဲ့ Password Hash ကို ရသွားပြီဆိုရင် Foreign Domain ထဲမှာ အဲဒီ Account အဖြစ် သုံးပြီး လှုပ်ရှားနိုင်သွားပါတယ်။

ဒီလို Hash ကို ဘယ်လိုရနိုင်သလဲဆိုရင် Forest တစ်ခုလုံးကို အပြည့်အဝ Compromise လုပ်နိုင်တဲ့ Attacker က **DCSync Attack** လုပ်ပြီး Trust Account ရဲ့ Password Hash ကို ထုတ်ယူနိုင်ပါတယ်။ Hash ကို ရသွားတာနဲ့ Attacker က Trust Account အဖြစ် Kerberos Authentication လုပ်နိုင်သွားပါပြီ။

Attack

ဒီနေရာမှာ အသုံးများတဲ့ Attack Technique က **Kerberoasting** ပါ။ ပုံမှန် Kerberoasting က Service Account တွေရဲ့ Service Ticket ကို တောင်းပြီး Offline Password Cracking လုပ်တာဖြစ်ပါတယ်။ Inter-forest Trust Attack မှာတော့ အထူးခြားဆုံးက Service Ticket တောင်းတဲ့ Account က လူသုံး Account မဟုတ်ဘဲ **Trust Account** ဖြစ်နေပါတယ်။ ဒါပေမယ့် Log တွေထဲမှာ ကြည့်ရင် အပြင်ပန်းအားဖြင့် Kerberoasting နဲ့ မထူးခြားပါဘူး။ “ဘယ် Account က Ticket တောင်းနေလဲ” ဆိုတာကို သေချာမကြည့်ရင် လွယ်လွယ်နဲ့ မသိနိုင်ပါဘူး။

```
alice-pc        Process Create    "C:\Users\alice\Desktop\Rubeus.e
                (rule: ProcessC   xe" asktgt /user:acquisition$ /d
                reate)            omain:aceresponder.local /rc4:43
                                  aec68656e12ba49bc784d8c64ac5f9 /
                                  dc:dc.aceresponder.local /ptt
```

ဒီ Scenario ထဲမှာ Attacker က Acquisition Forest ကို အပြည့်အဝ Compromise လုပ်ပြီး DCSync နဲ့ ACQUISITION$ Account ရဲ့ Hash ကို ရယူထားပါတယ်။ အဲ့ဒီ Hash ကို သုံးပြီး ACQUISITION$ အဖြစ် Logon လုပ်ကာ ACEResponder Forest ထဲက Service တွေအတွက် Service Ticket တွေကို တောင်းပါတယ်။ ဒါဟာ Inter-forest Trust ကို အသုံးချပြီး Forest တစ်ခုကနေ တစ်ခုထဲကို Pivot လုပ်သွားတဲ့ နည်းလမ်းဖြစ်ပါတယ်။

Test : Kerberoasting Results

How many service accounts did the attacker roast with the ACQUISITION$ user?

Tip: Check out the Intro to Detection Engineering module!

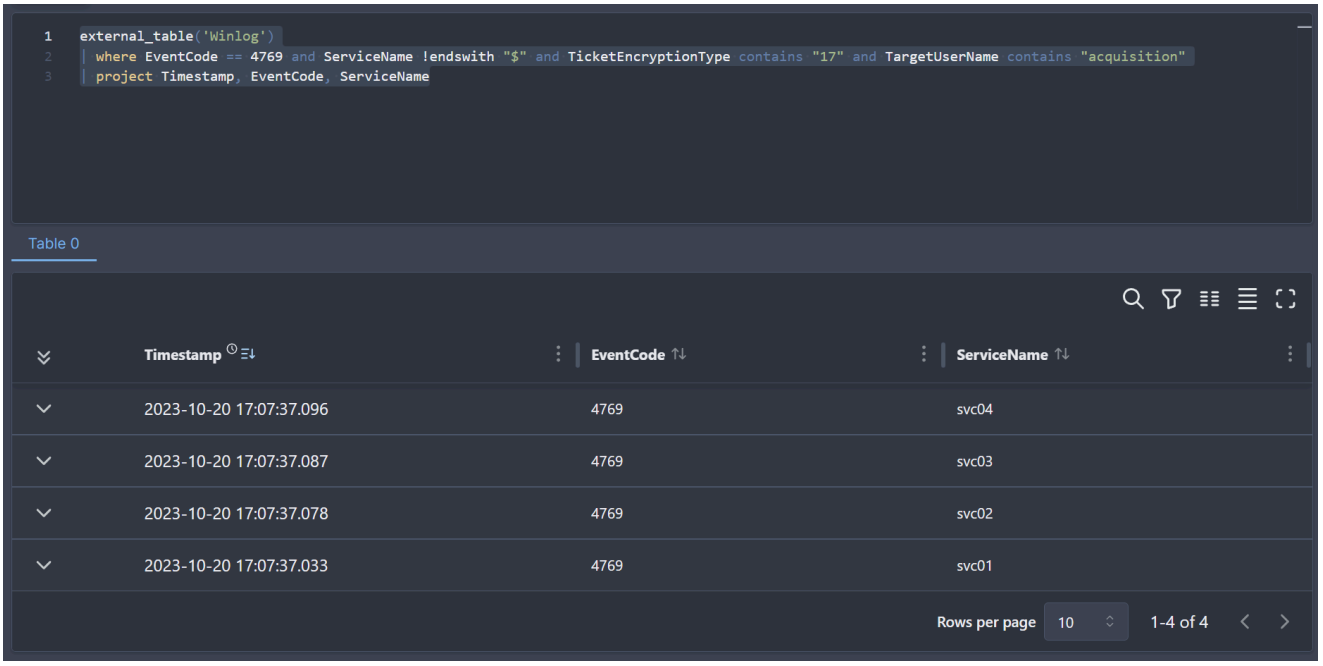
အရင်ဆုံး Kerberoasting ဆိုတာကို အတိုချုံး ပြန်ချိတ်ပါမယ်။ Kerberoasting က Domain ထဲမှာရှိတဲ့ Service Account တွေအတွက် Service Ticket (TGS) ကို တောင်းယူပြီး အဲဒီ Ticket ကို Offline Password Cracking လုပ်နိုင်အောင် ရယူတဲ့ Attack Technique ပါ။ Log အနေနဲ့ဆိုရင် Kerberos Service Ticket တောင်းတဲ့အခါ **Event ID 4769** ကို Domain Controller မှာ တွေ့ရပါတယ်။

ဒီ Scenario မှာ ထူးခြားတာက Kerberoasting လုပ်တဲ့ User က လူသုံး User မဟုတ်ဘဲ **ACQUISITION$** ဆိုတဲ့ Trust Account ဖြစ်နေတာပါ။ ဒါကြောင့် Defender အနေနဲ့ “ACQUISITION$ က ဘယ် Service Account တွေအတွက် Ticket တောင်းထားလဲ” ဆိုတာကို ရှာရပါတယ်။

```
external_table('Winlog')
| where EventCode == 4769 and ServiceName !endswith "$" and
```

```
TicketEncryptionType contains "17" and TargetUserName contains "acquisition"
| project Timestamp, EventCode, ServiceName
```

ပေးထားတဲ့ Query ကို အဆင့်လိုက် နားလည်အောင် ရှင်းပြရရင်၊
EventCode == 4769 ဆိုတာ Kerberos Service Ticket တောင်းတဲ့ Event တွေကိုသာ ရွေးထားတာဖြစ်ပါတယ်။
ServiceName !endswith "$" ဆိုတာက Computer Account (ဥပမာ SERVER တွေအတွက် Ticet မဟုတ်ဘဲလူသုံးSerice Account တွေအတွက် Ticet ကိုပုံစံချင်ဖြစ်တာကို သေချာအောင် Filter လုပ်ထားတာပါ။



ဒီ Query ရဲ့ Result မှာ ပြလာတဲ့ **ServiceName တစ်ခုစီ** က Attacker က Kerberoast လုပ်ခဲ့တဲ့ **Service Account တစ်ခုစီ** ကို ကိုယ်စားပြုပါတယ်။ အဲဒီ Result ထဲမှာ ServiceName ဘယ်နှစ်ခု ထွက်လာလဲဆိုတာကို ရေတွက်လိုက်ရင် “Attacker က ACQUISITION$ User နဲ့ Kerberoast လုပ်ခဲ့တဲ့ Service Account အရေအတွက်” ကို သိနိုင်ပါတယ်။
