welcome back to security onion essentials in this session we're going to look at our final workflow detection engineering now detection engineering is the process of developing technical means for uncovering malicious activity or even misconfigurations that could lead to malicious activity or compromise and really the first step in this process is detection gap is realizing that there is a gap in your detection now this could be from a formal process a gap analysis or more likely it's from an informal connection somewhere for instance from the lessons learned of a recent incident so once you've worked through and understood the detection gap itself the next step in the process is configuring the detection pipeline and this phase focuses on figuring out what data you need to get into security onions so that you can so that you can alert off of it right this could mean put standing up a new sensor in a segment in your network that wasn't there previously could also mean shipping windows event logs from another set of servers that you weren't shipping logs from before whatever whatever you need to do to to close that detection gap from a data perspective the underlying data that's what that configuring the detection pipeline is that could also mean parsing right making sure that data is parsed correctly as it gets into security onion so once we have visibility on the data itself now we can actually write and test our detection it could be uh writing a sierra cod snort rule maybe a zeke script or in what we'll see in just a minute writing a sigma signature uh for playbook um part of that is testing that detection and then once uh once that process has worked through and the test is succeeding moving that over to production all right and tuning right it's going to be extremely rare that you once you move your detection over into production that you're going to be able to let it sit for the next year and not have to tune at all um certainly you may need to tighten up the detection logic a little bit more or loosen it just depending on what you're seeing once you've actually moved it into production all right now this process uh can be implemented in a number of different ways inside security i mean like i mentioned you could use this process to uh to write a new sierra counter rule um but in this session we are going to be focusing on implementing this process specifically using a security onion playbook and there are three outputs from this process that are really critical for playbook the first one is an objective when we write a new detection specifically for playbook we want to be able to make sure that we have uh context around what we're doing and why what does this what does this detection look for why is it looking for it give me some give me some perspective some some context from the time that you've actually written the detection and put it into production maybe six months later when you're having to go back and do some further tweaking on it hopefully uh hopefully you've slept in that six months right um and so you may have uh forgotten exactly what you were doing and why um and or possibly you've moved on and transitioned off the team and there's someone else that is looking over the detection being able to have a clearly documented objective is going to be really key for your detection the second output from that process is the machine query and again this could look different but depending on the situation but what we're going to be looking at today is specifically using playbook which we need an elasticsearch query for that elastic search is what we use in the back end for security onion so we need the actual machine query that is going to implement that objective right so that's that machine query finally is the next steps once we actually have moved this detection into production and we have alerts that have been generated from the detection what are the next steps um how do i validate it uh how do i um you know how do i start remediating it um what are the next steps and i find this to be really really uh important because i'm sure we've all been there where we're looking staring at an alert we're not familiar with the malware family and the title of the alert and so then we try to dig into the rule itself and figure out what it's looking for then we may need to go do some research on a blog and figure out what exactly is going on here and what i need to do to go check to see if uh something has really been compromised right and so being able to put those that kind of information both in the objective and next steps i think is going to be really key for the analyst that has to work through these alerts to be able to quickly and efficiently triage and remediate as necessary so those are the three key outputs from that process we just talked about four four specifically for playbook so we're going to take objective the machine query and next steps and that's what makes up a play in playbook again we have this application called playbook and inside it are plays or detection plays and each of these plays has an objective the machine query that we need and the next steps now certainly there's other relevant metadata about the play but the three key pieces here are objective machine query and next steps i want to give credit where credit is due and these concepts and specifically playbook itself was inspired by a book that was published a couple years ago called crafting the infosec playbook feel free to check it out if these topics are of interest to you all right so enough theory let's go ahead and see what this looks like inside playbook in security onion all right so i'm starting out logged in to the security editing console and i'm going to come over here to playbook and we'll see that uh there are a number of things going on here on the far right hand side we have a way to filter some of the different plays right now it's on the default we're showing all the different plays and there are a lot of different plays that we already have uh created inside playbook for you um so we'll talk about that more in just a minute but we do pre-create a number of plays if we look at one of these uh there is a bit of metadata about the play but i want to focus on the three aspects that we were just talking about a second ago that first one is the objective of the play right here so network shares with loose access controls are common places that leak sensitive information this play proactively looks for newly shared resources that likely contain sensitive data follow-up will be needed to confirm that appropriate access control is in place all right so this gives us a little bit of context about what this play is looking for and why next is the machine query needed for the play and that's under the last alert config we see this is the actual elastic search query needed for this detection play uh if you're familiar with the last alert we do use a last alert in the background for playbook finally our next steps if we open up sigma which again we'll talk about in just a second scroll all the way down we'll see tasks there are two next steps check share permissions and contact system service owner uh we'll see where these come into play and why they're relevant in just a minute uh when we create our own play okay so those are the three key parts of a play now again there's lots of other metadata about the play the status the priority title author level playbook um but the three parts are that i just showed you to me are the most important if we come back to be clear all of these plays that we have pre-created not all of them have like next steps or or a very clear objective but i would urge you as you look at writing your own detection plays to really strive for those three key aspects specifically the objective and next steps so with that in mind and a quick overview of playbook let's go ahead and set up a scenario and walk through actually uh the process that we just talked about and the scenario is this uh in the aftermath of an incident it's become clear that your team missed the intrusion early on because of a detection gap specifically uh local accounts were being created on windows servers and being used for persistence as well as lateral movement what we're going to do is walk through that process with scenario and build out a new detection specifically for when local accounts are created on windows systems so the first one uh the first part that process is detection gap which we just talked about the second is uh developing your detection pipeline right we're making sure that we've got the data that we need in security onion i'm gonna assume for the sake of just the demo that we've we've got this already if i come over to hunt and i actually search for event code 4720 and i do have a result here so this is a windows event log from the security event channel event id 4720 that event id is generated when a local user is created on the system and so again for the sake of the demo we'll assume that we've now started uh shipping windows event logs specifically from our windows servers um and they're in security onion and they're parsed correctly if we drill down in here we'll see that we've got uh we've got data parsed out okay windows event id and some of the other data here okay so we've got our we've got our target log that we want to generate an alert when we see this right whenever a new local account is created or event id 4720 whenever we see that event id we want to generate an alert with our play so let's go over to playbook and click on sigma editor and i'm going to paste the play that i already created in here uh if you're not familiar with sigma or yaml this may look a little overwhelming i completely understand first of all uh sigma so sigma is what we use to generate plays if we look at our documentation just click on this documentation link that will bring you to creating a new play little blurb here about sigma rules from sigma's github repository sigma is a generic and open signature format that allows you to describe relevant log events in a straightforward manner the rule format is very flexible easy to write and applicable to any type of log file so that could mean zeke log that could mean wazoo windows event log sysmon log whatever it may be the main purpose of this project is to provide the structured form in which researchers or analysts can describe their once developed detection methods and make them shareable with others all right so another way that they put it uh in their documentation is that just like we have uh snort or sierra cotta rules and just like we have uh yara rules for files right sigma rules are for logs so it's the same type thought process here and certainly if you're not familiar with sigma and want to learn more you can read through our documentation you can also check out their repository which has some guides on uh the sigma specification and how to write sigma rules so coming back to our play here let me just briefly walk through what this looks like again sigma rules are written in yaml we do have syntax highlighting here uh the description is our objective all right so this play looks for new account creations on windows systems specifically for event id 4720 accounts can be used to maintain persistence these accounts are either created by the attacker or are legitimate like service accounts either way creation of accounts should be tracked and verified event id 4720 is generated when a user account is created so that's our objective now the actual machine query we need is uh written in this this log source and detection section right here again this is specific to sigma i'll show you what happens in just a minute um but this is actually what generates you'll see we talked about event id 4720. i'll show you in just a second why this matters all right finally our third is next step so tasks when this play generates alerts um what should the analyst do so the first one is does the username conform to our naming standard uh so does you know some some places have a particular naming schema for service accounts for admin accounts for normal user accounts if it doesn't conform to that then continue to step two go ahead and contact this system service owner for remediation so this may be a tip-off that this may not be a legitimate account or if it is legitimate it needs to be checked to you know renamed according to uh the internal schema there's certainly a lot more that you could do here right but between the tasks and the objective that should give the analysts a starting point for what this play is looking for why and how to analyze it once uh once we see an alert created so this is great josh but how in the world do we get from this to an actual play and get alerts from it i'm glad you asked we click on convert and uh what's going to happen in the background is playbook is going to take this log source and detection clause and going to convert it with our field mappings into this elastic search query specifically we're looking for that winlog.channel security and event.code security 4720. now we could change this just to show you 4721 click on convert and uh that'll change the query but we do want to keep it at 47.20 so let me click convert and we need to test that our target log will come up when we use this query so i'm going to click on copy and come over to hunt and i'll paste this query in hunt hit enter and we should see the two logs that i have uh in elasticsearch which i do which this is good this means that this uh that this detection is valid uh if i change that to 4721 i will get no results because i don't have any of those logs in uh in my environment right so we know this detection is going to alert on the logs that we have in the environment so from now i can go back to playbook i can say create play from sigma and it's going to take just a couple seconds to go ahead and convert that over again and then create the play in playbook you can see that playbook pulled out the title and the author and some of the other metadata about the play i also pulled out the objective the last alert configuration which looks familiar and finally we have our next steps down here under tasks which again we'll see in just a minute why that matters all right so we have we've we've done our third phase in the detection engineering process right we've created and tested our detection now it's time to actually move this into production because right now we just created the play but you can see that the status is draft that means that it's not actually going to get any alerts if it sees any of those event ids so we're going to click on edit change the status from draft to active click on submit and what this does is this moves that last alert configuration into production and now security engine is going to start running that query last alert runs every three minutes so it's going to start looking for those logs every three minutes all right now again for this demo i've got alert already generated so i can show you what that looks like if you would implement this in production and you get an alert generated from this play here's what that will look like if i come over to alerts and scroll down you see that we have event up module we have our ciracada our osakawasu alerts if you scroll all the way down we'll see that we have a playbook alert so user account created on windows we'll drill down into that there's some some metadata about the alert in here we can also or it gives us the event itself right the data itself about the event it also gives us a link to the play so we could right click and go to that play and it would bring up this oops excuse me would bring this up so we could look into more about what's going on why was this alert generated if we want to actually create a case this is an issue i need to go check on this we can click on the escalate button and that's going to create a case in the hive this is where tasks come into play when we have that case created just right here in the hive user account credit on windows if we click on that case we'll see that we already have tasks that need to be done for this case so i'll be able to say i'm going to start this this task does a username conform to a our naming schema um i can add my task log no it doesn't i need to go ahead and contact the system owner and go from there so again this this is really to me this is really important excuse me because you get to you get the whole chain of events here right i've created the play i've got the objective um and then once i get the alert i can create a case from it and i've actually got all that data right here in my case i've got context i've got the machine query i've got the event itself that generated the alert and i have next steps so from the endless perspective i can drop right in here to the case and start quickly and efficiently working this case and doing what i need to do to keep my organization and my users safe so that is in a real real brief uh walk through both detection engineering and playbook so if we kind of come back here full circle excuse me we talked about the detection engineering process and then how the outputs of those process the objective the machine query and the next steps are really key components to a detection play in playbook then i came over here i demoed playbook for you i actually created a play for event id 4720 for a windows environment we've moved that play into production i showed you what an alert looks like escalated that to the hive and showed you how those next steps come into that case in the hive all right so lots of moving pieces here but hopefully that gives you kind of an idea of how this all works together and i think how powerful it will be in with you in your environment so that's it for detection engineering and our final workflow workflow number three thanks again and i'll see you on our next session