

Attack Trees: Door de bomen de bedreigingen zien

Sjouke Mauw en Martijn Oostdijk

Attack trees

Een van de belangrijkste stappen in het beveiligen van systemen is het uitvoeren van een risicoanalyse. Perfecte beveiliging bestaat niet en dus moeten keuzes gemaakt worden: welke assets zijn kritisch genoeg om extra beveiligd te worden?

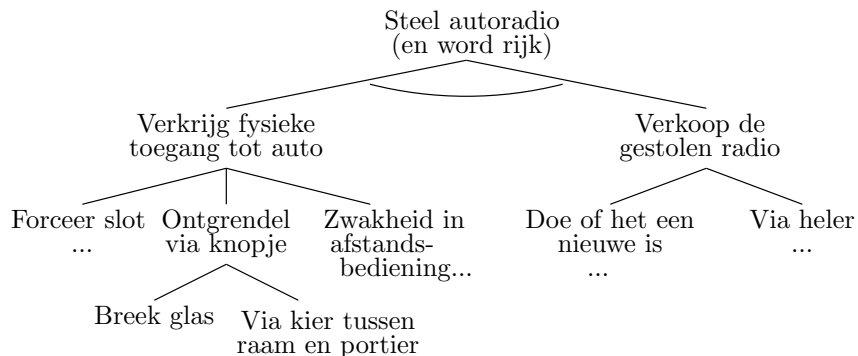
Het is dan ook niet verwonderlijk dat standaarden zoals de “Code voor Informatiebeveiliging” en z’n Britse broertje “BS7799” als één van de eerste stappen aanraden om zo’n analyse uit te voeren. Een risicoanalyse is een proces waarin de volgende drie vragen beantwoord worden:

1. Welke bedreigingen zijn er?
2. Hoe groot is de kans dat zo’n bedreiging werkelijkheid wordt?
3. Wat is vervolgens de impact?

Hoe deze vragen te beantwoorden? Daar zegt zo’n standaard wijselijk niet zo veel over. Alleen maar *dat* het moet gebeuren en wel op een formele manier.

Nu is het uitvoeren van een risicoanalyse ook lang niet eenvoudig: er moeten kansen geschat worden om iets over het risico te kunnen zeggen. Ook de te verwachten impact kan van allerlei toevalligheden afhangen. Het blijft giswerk, alhoewel er voor zowel kwantitatieve als kwalitatieve risicoanalyse allerlei methodieken ontwikkeld zijn. Maar het allermoeilijkste is het inschatten van welke bedreigingen er überhaupt zijn.

Een manier om de bedreigingen toch systematisch in kaart te brengen, *attack trees*, werd voorgesteld door Bruce Schneier in zijn artikel uit 1999 in Dr. Dobb’s journal [2]. Dit formalisme werkt goed om een beeld te krijgen van de mogelijkheden van een aanvaller. Schneier kiest duidelijk voor het perspectief van de aanvaller: de aanvaller wil een bepaald doel bereiken (weergegeven in de wortel van de boom) en het doel kan op meerdere manieren bereikt worden (een nieuwe vertakking in de boom). Soms is zelfs al ongeveer duidelijk welke deeldoelen tezamen bereikt moeten worden om een doel te bereiken (een aantal “gebundelde” takken). Een voorbeeld van een eenvoudige attack tree is gegeven in Figuur 1.



Figuur 1: Eenvoudige attack tree.

Bovenin de boom (bij de wortel, merk op dat bomen in dit soort verhalen altijd van boven naar onder groeien) blijven de doelen vrij abstract. Onderaan (bij de bladeren) staan de concrete doelen die weinig implementatiekeuze meer bevatten. Een boogje bij een vertakking betekent dat alle deeldoelen bereikt moeten worden. Als er geen boogje staat mag de aanvaller kiezen.

Het grote voordeel van de attack tree methode is dat het gaat om een semi-formele methode. De boom drukt een formele relatie tussen verschillende aanvallen uit, maar de aanvallen zelf zijn uitgedrukt in natuurlijke taal. De gebruiker kan zelf bepalen welke mate van detaillering voor zijn toepassing geschikt is. Dit, samen met de intuïtieve notatie, maakt attack trees tot een eenvoudig te gebruiken methode.

Het opstellen van een attack tree is op zichzelf vaak al een nuttige exercitie. Bij het opstellen van een attack tree door een groep security experts komen vaak verrassende zwakheden boven water die in eerste instantie over het hoofd gezien waren. Het is natuurlijk in het algemeen niet mogelijk om alle zwaktes van een systeem te bepalen, waardoor het vaak niet mogelijk is om een volledige attack tree te maken.

Het begint pas echt interessant te worden als er eenmaal een concrete boom op papier staat. Dan kan er namelijk “aan gerekend gaan worden”. Schneier heeft hier een tweetal suggesties voor:

- **Attributen.** Als je wilt weten wat de kosten zijn die een aanvaller moet maken om de in de wortel beschreven aanval uit te kunnen voeren, dan kun je volstaan met het schatten van de kosten van de bladeren. De kosten die behoren bij de daarboven liggende knopen kunnen nu (automatisch) afgeleid worden uit de kosten van de bladeren. De kosten zijn dan een *attribuut*. Andere voorbeelden van attributen zijn: wel of niet speciaal gereedschap nodig, risico voor de aanvaller, additionele impact voor de verdediger, etc.
- **Projecties.** Als een aantal attributen doorgerekend is, kan bepaald worden welke deelboom overblijft als we alleen die aanvallen bekijken waarvan de waardes acceptabel of realistisch zijn. Dit maakt het mogelijk om verdedigingen te ontwerpen die alleen gericht zijn op de relevante aanvallen.

Schneier’s artikel illustreert bovenstaande operaties aan de hand van een aantal voorbeelden. En alles lijkt helder.

Fundamenten

Je kunt je afvragen hoe handig die attack trees nou werkelijk zijn. Op de voorbeeldjes van Schneier werkt het allemaal prima, maar werkt het in het algemeen ook?

Een eerste stap om dit te onderzoeken is het geven van een formele semantiek. Wat betekent zo’n boom nou eigenlijk precies? In [1] voorzien we attack trees van zo’n semantiek. Een verrassende ontdekking was dat het niet slim is om attack trees te definiëren als bomen met en-of-knopen, zoals ze vaak omschreven worden, maar als *bundels* van takken. Verder definiëren we transformaties om bomen, zonder verlies van betekenis, om te schrijven naar eenvoudigere bomen.

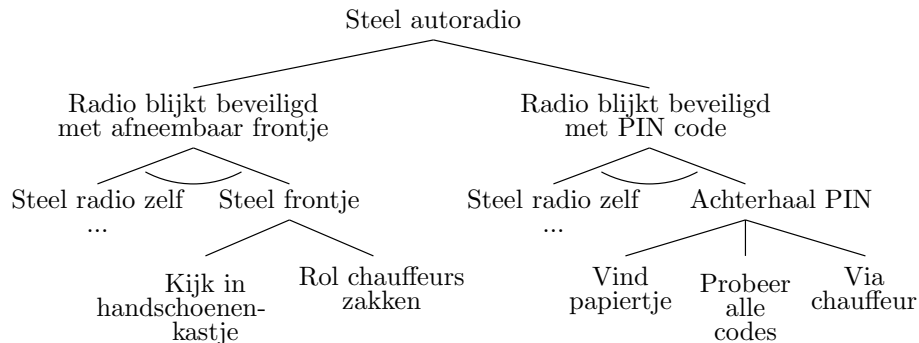
Op basis van de gegeven semantiek kunnen we vervolgens heel precies aangeven onder welke voorwaarden de door Schneier voorgestelde operaties mogen. De voorbeeldjes van Schneier werken “toevallig” goed, en bij het projecteren houdt hij “toevallig” altijd een boom over. In [1] staat uitgelegd waarom dat goed gaat.

Uitbreidingen

Naast onderzoek naar de fundamenten van het formalisme kijken we ook naar *hoe* attack trees gemaakt en gebruikt kunnen worden. Door middel van wat grotere case studies proberen we te achterhalen hoe je, met een groep stake-holders, zoveel mogelijk aanvallen boven tafel kunt krijgen. In projecten (soms uitgevoerd door studenten) kijken we naar attack trees voor elektronisch stemmen, betaalautomaten, social engineering, het biometrische paspoort, etc.

Bij het uitvoeren van zulke case studies vallen twee dingen op. Ten eerste, tool-support is hard nodig. Op papier ben je constant bezig om de boom te herbalanceren, nog even een sub-boompje ergens tussen te proppen, etc. We denken hierbij vooral aan tool-support voor de constructie van attack trees, waarbij de nadruk ligt op construeren, transformeren en projecteren. Het ontwerpen van tools voor de analyse van attack trees en de uitbreiding naar de andere facetten van een risicoanalyse zal later een rol spelen.

Ten tweede, tijdens het verzinnen van aanvallen ben je bijna automatisch al bezig met het verzinnen van mogelijke maatregelen die de aanval kunnen keren (of in ieder geval ervoor kunnen zorgen dat relevante attributen gunstige waardes hebben, vanuit het standpunt van de verdediger). Soms sluipen dat soort overwegingen ongemerkt in de attack tree, zoals bijvoorbeeld in Figuur 2.



Figuur 2: Verkeerd soort keuze in attack tree.

Er klopt iets niet aan deze boom: niet de mogelijkheden van de aanvaller, maar juist mogelijke maatregelen van de beveiliging bepalen welke deelboom van toepassing is.

Een formalisme waarin zowel vanuit het standpunt van de aanvaller als vanuit het standpunt van de verdediger naar mogelijke aanvallen gekeken wordt, *attack/defense trees*, lijkt een betere oplossing om met dit probleem om te gaan. Het verzinnen van mogelijke maatregelen behoort natuurlijk niet meer tot het domein van de risicoanalyse. Hoe *attack/defense trees* er precies uit gaan zien, en wat voor soort operaties op zulk soort bomen mogelijk zijn, is onderwerp van toekomstig onderzoek.

Conclusie

Attack trees lijken handig voor het modelleren van bedreigingen. Het opstellen van een boom is op zichzelf al nuttig. Het formalisme is informeel te gebruiken (hoewel enig inzicht op security gebied noodzakelijk is), maar heeft toch formele fundamenteen waardoor serieuze tool-support mogelijk lijkt.

Er valt nog genoeg interessant onderzoek te doen naar uitbreidingen van attack trees. Hierbij ons verlanglijstje:

- Attack/defense trees ontwerpen.
- Ontwikkelen van een prototype tool.
- Het nut van attack trees in de praktijk onderzoeken. (Door case studies te doen.)
- Kijken of attack trees gebruikt kunnen worden voor systemen die nog ontworpen en gebouwd moeten worden?
- Ontwikkelen van een bibliotheek van herbruikbare attack trees. (Dit werpt weer allerlei nieuwe vragen op over de betekenis van de bomen.)

Dit alles en meer, hopelijk in toekomstig werk.

Over de auteurs

- Dr. Sjouke Mauw is universitair hoofddocent binnen de Formele Methoden groep aan de Technische Universiteit Eindhoven.
- Dr. Martijn Oostdijk is universitair docent binnen de Security of Systems groep aan de Radboud Universiteit in Nijmegen.

Referenties

- [1] Sjouke Mauw and Martijn Oostdijk. Foundations of attack trees. In *Proc. 8th Annual International Conference on Information Security and Cryptology, ICISC'05*, 2006. To appear.
- [2] Bruce Schneier. Attack trees: Modeling security threats. *Dr. Dobb's journal*, December 1999.