Deze keer een interview met Victor van der Veen, promovendus aan de Vrije Universiteit Amsterdam over Android in aanloop naar zijn presentatie op de Black Hat Sessions.

Victor van der Veen is promovendus aan de Vrije Universiteit Amsterdam, in de vakgroep van Herbert Bos. Tijdens de Black Hat Sessions op 23 juni zal hij een technische lezing verzorgen over Android. In dit (technische) interview laat Victor zijn licht schijnen over de vragen die we hem in aanloop naar de conferentie vast stelden.

Kun je iets zeggen over het promotieonderzoek dat je aan de VU uitvoert?
De projecten waar ik vooral mee bezig ben geweest, gaan over Control-Flow Integrity (CFI) op binary-niveau. CFI is een verdedigingstechniek tegen geavanceerde aanvallen zoals Return Oriented Programming (ROP). Om een ROP-aanval te laten slagen, misbruikt een aanvaller bijvoorbeeld een "buffer overflow"-kwetsbaarheid om controle over een programma over te nemen. Bij een ROP-aanval injecteert de aanvaller zelf geen code, maar maakt hij gebruik van bestaande instructies van het programma [die reeds in het werkgeheugen zijn geladen, red.]. Door deze (blokken van) instructies in een bepaalde volgorde aan te roepen, neemt hij de controle over. CFI stopt zo'n aanval door te forceren dat instructies die een programma naar een andere locatie laten springen (control-flow instructies), alleen naar legitieme locaties (zoals oorspronkelijk bedoeld door de ontwikkelaar) kunnen springen. Een CFI-oplossing op binary-niveau betekent dat we geen broncode van het originele programma nodig hebben om het te beschermen.

De CFI-projecten waar ik aan heb gewerkt heten PathArmor en TypeArmor. PathArmor maakt gebruik van recente features in Intel processors om programma's te beschermen met een sterke variant van CFI: zogenaamde "context-sensitive CFI"[1]. Door gebruik te maken van hardwarefeatures heeft PathArmor een lage run-time overhead. TypeArmor focust op zogenaamde forward-edge control-flow instructies (indirect call instructies, in het bijzonder; instructies die je bijvoorbeeld ziet wanneer je een functiepointer aanroept). Met TypeArmor is het ons gelukt om een recent aanvalsmodel te stoppen (namelijk: Counterfeit Object Oriented Programming, oftewel COOP) [2].

Hoe is mobiele malware te herkennen?
Ik denk dat het tegenwoordig moeilijk is om mobiele malware te herkennen. Ik zal in mijn talk een demonstratie geven van (door ons geschreven) mobiele malware en ik denk niet dat een gebruiker deze malware met het blote oog kan herkennen. Ook automatische analyse kunnen we omzeilen; onze malafide app heeft een paar maanden in de Play-store gestaan en werd pas verwijderd nadat ik het hoofd van Android Platform Security een demonstratie filmpje heb laten zien.

Wat is een groter probleem: kwaadaardige applicaties die de gebruiker simpelweg om te veel rechten vragen, of applicaties die gebruik maken van kwetsbaarheden in het besturingssysteem? (Stagefright etc.)
In principe zou een gebruiker apps die om te veel rechten vragen gewoon kunnen weigeren tijdens installatie, en zou je zeggen dat kwetsbaarheden in het besturingssysteem een groter probleem zijn. Ik verwacht echter dat bestaande malware vaker gebruik maakt van een kwetsbare gebruiker (die alle permissies blindweg accepteert) dan van een kwetsbaar besturingssysteem.

Helpen mobiele anti-viruspakketten om gebruikers te beschermen tegen malware?
Slechts tot een zekere hoogte. Ik zou er niet blind op vertrouwen: ze kunnen veelal geen nieuwe malware detecteren.

Zie je trends in mobiele malware?
Ik heb me hier al een tijdje niet in verdiept, maar tot op heden staan vooral third-party markets erom bekend malware aan te bieden. Met name China heeft hier last van.

Gedurende je vorige onderzoeksprojecten ben je o.a. bezig geweest met geautomatiseerde analysetools voor Android-applicaties, zoals Andrubis. Hoe goed zijn deze tools in het analyseren/herkennen van malware, en hoe kunnen deze bedrijven helpen om veiliger te worden?
Andrubis geeft iedere app een rating tussen 0 (goedaardig) en 10 (kwaadaardig) en met de juiste threshold zijn de resultaten 'goed' te noemen. Van een sample set van 15.000 malware samples werd 98% correct gedetecteerd als kwaadaardig. Helaas is Andrubis ondertussen al aardig verouderd en worden grote apps of apps die gebruikmaken van nieuwere APIs niet ondersteund. Een bedrijf kan Andrubis gebruiken om het kaf van het koren te scheiden, maar er kunnen geen garanties gehangen worden aan of een app daadwerkelijk goed- of kwaadaardig is.

Hoe groot is het probleem van banking-trojans op mobiele devices, waar je onderzoek naar gedaan hebt? En hoe kan een gebruiker zich hiertegen beschermen?
Ik weet niet exact hoeveel verlies banken op dit moment lijden door banking-trojans, maar Eurograbber[3] is een goed voorbeeld van hoe fout het kan gaan: 36 miljoen euro gestolen.

Ik weet niet exact hoeveel verlies banken op dit moment lijden door banking-trojans, maar Eurograbber is een goed voorbeeld van hoe fout het kan gaan: 36 miljoen euro gestolen

Helaas heb ik geen goed nieuws en werkt Google tot op heden niet mee aan een manier om onze aanval te stoppen. Een radicale tip is het loskoppelen van je Android-account, met alle nadelen van dien, zoals geen Gmail meer op je telefoon.

Cryptolocker/ransomware is een steeds groter probleem voor traditionele computersystemen en netwerken. Verwacht je dat hier ook mobiele varianten op zullen komen, en waarom?
De mobiele varianten zijn er al, maar het is de vraag of dit net zo'n groot probleem gaat worden als op de desktop. Het lijkt erop dat Google er sinds Android 4.4 voor zorgt dat apps geen files kunnen verwijderen buiten hun eigen directory op de sdcard. Ze konden al niet bij normale data-directories van andere apps (niet op de sdcard), wat betekent dat ransomware alleen iets kan wanneer het onder root-rechten draait. Ransomware zou dus een root-exploit moeten bevatten; daar zou het bijvoorbeeld kingroot (http://www.kingroot.net) voor kunnen gebruiken.

Recentelijk heb je onderzoek gedaan naar kwetsbaarheden in tweefactorauthenticatie (2FA) wanneer deze op een mobiel device plaatsvindt. Kun je een korte samenvatting geven van je bevindingen?
We gaan uit van de situatie dat een aanvaller controle heeft over de browser van zijn slachtoffer. In deze situatie zorgt 2FA ervoor dat een aanvaller geen operaties kan uitvoeren die door 2FA beschermd worden (zoals het overmaken van geld van rekening A naar B). We hebben twee 'kwetsbaarheden' misbruikt om vanuit een geïnfecteerde browser ook controle te krijgen over de telefoon van een slachtoffer om zo phone-based 2FA te kraken. De eerste is de remote-install-feature van Google Play: je kunt een app installeren vanuit je browser door op de "Install"-button te klikken, en hoeft hierbij geen interactie uit te voeren op je telefoon (het accepteren van de permissies wordt in de browser voltrokken). Het tweede issue is de mogelijkheid om een app te activeren door op een link te klikken. Standaard is een app na installatie inactief; de app wordt pas actief zodra je deze opent, bijvoorbeeld door op het icoon te klikken of door op een speciale link te klikken die gekoppeld is aan de app. Omdat we controle hebben over de browser, en omdat alles tegenwoordig gesynchroniseerd wordt, kunnen we vanuit de browser bijvoorbeeld bookmarks van het slachtoffer overschrijven zodat ze, als deze geopend worden op een telefoon, redirecten naar onze app en deze activeren. De malafide app kan vervolgens sms-berichten afluisteren en TAN-codes doorsturen naar de aanvaller zodat hij midden in de nacht je bankrekening kan Leegtrekken. Dit onderzoek is trouwens ook onder de aandacht gekomen bij Slashdot[4, 5].

Zie je goede alternatieven voor de gebruikelijke manieren van 2FA?
Het is belangrijk om een tweede factor te gebruiken die los staat van de eerste factor (je PC/laptop). Een cardreader is daarom veiliger dan je smartphone (omdat de smartphone kan worden geïnfecteerd als je die koppelt met je PC/laptop).

Het Android platform wordt steeds meer gebruikt in apparaten, anders dan traditionele telefoons en tablets. Zie je hier gevaren?
Het is inderdaad te verwachten dat de komende jaren koelkasten en andere Internet of Things (IoT)-apparaten gehackt gaan worden. Hoe dit precies zal verlopen, durf ik niet te voorspellen, maar onze hack laat zien dat het verbinden van apparaten en synchroniseren van gegevens nieuwe aanvalsmodellen mogelijk maakt.

Victor van der Veen

Victor is a PhD candidate in the System and Network Security Group at the VU University Amsterdam where he also obtained his MSc. degree in Computer Science in August 2013. Victor is currently under the supervision of prof. dr. ir. Herbert Bos.
His research focuses on - but is not limited to - malware on smartphones and is part of the Dutch-American Project Arrangement about cooperative research and development on cybersecurity. This means that he will spend a significant amount of time at the University of California Santa Barbara, where he will be advised by prof. dr. Christopher Kruegel. Besides mobile malware, Victor is interested in (low-level) system topics that enhance system security, as well as reverse engineering and analyzing malicious code. Aside from doing research on these topics, he also enjoys implementing related features in real systems.

His personal website, that includes a list of publications, can be found at http://vvdveen.com/.

1 http://vvdveen.com/publications/PathArmor.pdf
2 http://vvdveen.com/publications/TypeArmor.pdf
3 http://www.bankinfosecurity.com/interviews/darrell-burkey-i-1730/op-1
4 https://it.slashdot.org/story/16/04/08/1735240/anywhere-computing-makes-2fa-insecure-on-ios-and-android
5 https://news.slashdot.org/story/16/04/10/237215/academics-claim-google-android-2fa-is-breakable