poročilo

MICROSOFTOVA 20. NT KONFERENCA 2015

NT konferenca velja za največjo slovensko tehnološko konferenco, ki jo organizira podjetje Microsoft Slovenija. Letošnja konferenca je bila jubilejna, saj se je odvila že dvajsetič. Potekala je od 18. do 20. maja 2015 v Portorožu. Udeležilo se je je 1500 udeležencev, sodelovalo je več kot 150 predavateljev.

Konferenca je potekala v naslednjih desetih vzporednih vsebinskih sklopih, na katerih so s svojimi zanimivimi prispevki sodelovali tako domači kot tuji strokovnjaki:

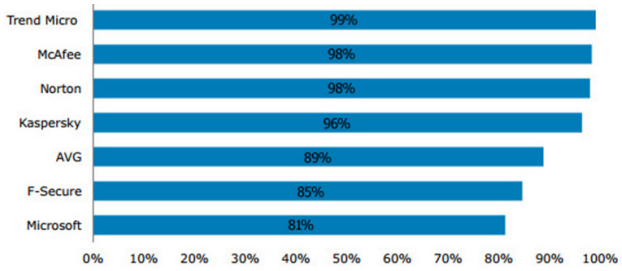
- SQL Server,
- Windows,
- varnost in identitete,
- strežniki in upravljanje,
- spletni razvoj,
- razvojna orodja in arhitektura,
- primeri iz prakse,
- razvoj aplikacij,
- poslovne rešitve in
- produktivnost.

Sodelavci Instituta informacijskih znanosti (IZUM) smo se konference udeležili drugi in tretji dan. V nadaljevanju predstavljamo nekaj vsebin in vtisov s predavanj ter razmišljanj, ki so se oblikovala ob tem dogodku.

PATRICK DALVINCK: INSTANT ON CLOUD SECURITY

Slika 1: Varnost v oblaku (Vir: Trend Micro, 2015)

Predavanje z naslovom *Instant on cloud security* je imel Patrick Dalvinck, Microsoftov partner in podpredsednik podjetja Trend Micro za območje Evrope. Predavanje je temeljilo na predstavitvi njihovega najnovejšega izdelka za zaznavanje virusov in škodljivega programja (angl. *malware*) ter za zaščito pred napadi in zagotavljanje varnosti v oblaku.



Slika 2: Povprečni delež blokad na prenos (Vir: Trend Micro, 2015)

Pokazal nam je, kako poskušajo pri podjetju Trend Micro zaščititi svoje uporabnike pred napadi hekerjev in podobnim.

Product	Hours
Trend Micro	0.24
McAfee	0.36
Norton	0.43
Kaspersky	11.36
Microsoft	20.23
AVG	24.80
F-Secure	32.36
Average	12.82

Slika 3: Povprečni čas za dodajanje blokad (Opomba: Povprečni čas, potreben za dodajanje detekcije novih groženj, je pomembno merilo.) (Vir: Trend Micro, 2015)

Razložil je, kako se hekerji običajno lotijo svojih žrtev z nadležno pošto (angl. *spam*), povezavami do sumljivih spletnih strani itd., tako da smo dobili občutek, kako ta postopek deluje. Glavni problem je žal še vedno nevednost običajnega uporabnika. Velik problem je v tem, da nekateri ljudje brezglavo klikajo in ne pogledajo ali preberejo, kaj se dogaja oz. kaj določena zadeva (npr. e-sporočilo) zahteva od njih. Če bi bili ljudje bolj pozorni in previdni, bi bilo veliko manj okužb računalnikov in napadov hekerjev.

JASMIN AZEMOVIĆ: CSI SQL SERVER

Sodobni informacijski sistemi, kot so e-učenje, e-uprava, e-univerza, e-volitve, e-bančništvo in e-zdravje, se pogosto

zlorabljajo z nepravilno spremembo podatkov. To dejstvo nas sili k temu, da ponovno preučimo naše varnostne ukrepe in poskusimo najti način, da jih izboljšamo. Dokazovanje računalniškega kriminala je zelo zahteven in zapleten proces, ki temelji na digitalnih dokazih in zbiranju podatkov, forenzični analizi in postopku preiskave.

Digitalni dokazi, zbiranje podatkov in forenzična analiza sistemov baze podatkov so zelo specifične in zahtevne naloge. Predvsem je ključno, kako dokaze interpretiramo in uporabljamo, da pri tistem, ki jih zbira, ni konflikta interesov. Prikazano je bilo, kako lahko, ne glede na varnostne kopije oz. stare podatke, ugotovimo, ali je administrator baze podatkov, ki ima za neko bazo dodeljene vse pravice, spremenil zapis z običajnim stavkom SQL UPDATE. Za odkrivanje takšnih zlorab v aplikacijo vgradijo enosmerno funkcijo *hash* in ko nekdo priredi neki podatek zunaj aplikacije (npr. zamenja priimek osebe), se stolpec s funkcijo *hash* seveda ne spremeni. Ko na spremenjenem zapisu še enkrat uporabimo funkcijo *hash*, nam ta ne vrne več enake vrednosti, kot je bila v izvirniku, kar pomeni, da je nekdo prirejal podatke. Seveda se postopek tukaj šele začne. Kdo je podatke zares spremenil, se ugotavlja preko drugih orodij, se pa s tem krog zoži.

Predavatelj prihaja iz Bosne in Hercegovine (zaposlen je na Fakulteti za informacijsko tehnologijo v Mostarju in se med drugim ukvarja s kriptografijo in forenzično analizo); povedal je, da imajo v BiH na tem področju zelo kaotično stanje in da jih čaka še veliko dela.

MADS KRISTENSEN: MODERN WEB TOOLING IN VISUAL STUDIO 2015 IN JERNEJ KAVKA: VISUAL STUDIO EVERYWHERE – LINUX, OSX (IN WINDOWS, SE RAZUME)

Aplikacije ASP.NET 5 bo možno zaganjati na platformah Windows, OSX in Linux. Spletni obrazci (angl. *web forms*) iz preteklih različic niso več podprti, obstoječo kodo bo treba prilagoditi ASP.NET MVC (MVC 6), ki zdaj uporablja univerzalni razred, namenjen kontrolnikom MVC in Web API. AngularJS je najbolje podprto ogrodje za gradnjo odjemalcev (podprti so tudi Bootstrap, jQuery, Polymer), za komunikacijo z MVC 6 se uporablja REST.

Ukinjena je podpora za Visual Basic.Net, APS.NET 5 podpira samo jezik C#. Vpeljani so novi Unit-testi, imenovani xUnit.net, ki so prilagojeni za testiranje spletnih aplikacij. Aplikacije lahko uporabijo tudi najnovejši prevajalnik, napisan povsem na novo v jeziku C# – Roslyn (prejšnje različice so bile v C++). Koda se prevaja dinamično, med pisanjem. Na ta način želijo zagotoviti čim manj prekinitev pri delu, podobno kot je to pri pisanju interpreterske kode. Trenutni povprečni čas, potreben za zagon aplikacije ASP.NET med razvojem, ki znaša več kot sedem sekund, želijo skrajšati na eno sekundo.

Prevajalnik Roslyn omogoča programsko spremljanje in vplivanje na postopek prevajanja. Prevajanje je lahko uporabljeno kot del programskih rešitev. Na ta način se C# zelo približa uporabnosti dinamičnih programskih jezikov, kot sta Python in Ruby. Novi prevajalnik bo podpiral samo C# in VB.Net.

Pomembna novost je, da bo izvajalsko okolje .Net Framework predstavljalo del namestitvenega paketa aplikacije ASP.NET. Tako se ne bodo več pojavljale situacije, ko bi neka redna varnostna posodobitev izvajalskega okolja lahko povzročila nestabilno delovanje aplikacij. Zagotovljeno bo, da bodo razvojna, testna in produkcijska okolja tekla na enakih izvajalskih okoljih. Hkrati bo možno v namestitveni paket dodati samo tiste module izvajalskega okolja, ki so nujni za delovanje aplikacije. Posledično se bo povečala odzivnost aplikacij (prav tako bo hitrejši tudi prvi zagon) in hkrati zmanjšala poraba pomnilnika.

Aplikacije ASP.NET 5 lahko razvijamo tudi v Visual Studio Code. Gre za okrnjeno različico razvojnega okolja Visual Studio, ki deluje (bo delovalo) na različnih platformah, kot so OSX, Linux in Windows. Code je predvsem zelo dober urejevalnik (angl. *editor*), ki ima vgrajeno celotno podporo za gradnjo (intelliSense, parameter hints, inline errors, goto definition, find references, rename symbols …) in razhroščevanje aplikacij ASP.NET v jeziku C# in TypeScript (verzija JavaScripta s podporo razredov in vmesnikov).

MIKE MENGELL: ANGULAR FOR THE ENTERPRISE AND ITS SYNERGY WITH MICROSOFT TECHNOLOGIES IN GETTING THE MOST OUT OF CORDOVA AND ANGULAR

Mike Mengell, razvijalec aplikacij, ki ga zelo zanima tudi JavaScript, je imel zanimiva in koristna predavanja o razvoju enostranskih spletnih strani in aplikacij z uporabo orodja Visual Studio 2015. Na prvem predavanju z naslovom "Angular for the enterprise and its synergy with Microsoft Tehnologies" je predstavil ogrodje JavaScript AngularJS za gradnjo bogatih enostranskih spletnih aplikacij. Na začetku je povedal, da je AngularJS trenutno najpogosteje uporabljeno ogrodje JavaScript in da ima zelo dobro dokumentacijsko podporo. Nato je prikazal njegove glavne sestavne dele, kot so usmerjevalnik, direktiva, storitev, tovarna itn. Na praktičnih primerih je prikazal, kako uporabni funkcionalnosti sta vrivanje odvisnosti in dvostransko povezovanje podatkov med pogledom in modelom. Toplo je priporočal tudi uporabo vedenjsko usmerjenega razvoja spletnih strani (BDD) z uporabo ogrodja Jasmine. Na osnovi praktičnega

primera je prikazal, kako izvesti test in na kaj je pri tem treba biti pozoren. Poskušal je podrobneje predstaviti tudi rešitve SignalR, WebAPI in TypeScript, a je zmanjkalo časa, tako da nas je seznanil le s tem, za kaj se katera od naštetih rešitev uporablja.

Slika 4: Utrinek s predavanja M. Mengella

Tudi naslednji sklop predavanj z naslovom "Getting the most out of Cordova and Angular" je bil zelo kakovosten. Mengell je predstavil platformo Cordova za gradnjo avtohtonih (angl. *native*) aplikacij na osnovi HTML, CSS in JavaScript. Prikazal je, kako lahko uspešno uporabimo AngularJS pri gradnji avtohtonih aplikacij znotraj platforme Cordova ter kako lahko sočasno razvijamo aplikacije za operacijske sisteme iOS, Android in Windows Mobile. Na osnovi številnih praktičnih prikazov smo lahko videli, kako lahko preko platforme Cordova dostopamo do strojne opreme mobilnih naprav in kako jo uporabimo znotraj spletnih aplikacij. Predstavil nam je odzivne čase aplikacij, grajenih v Cordovi, ter povedal, kateri od teh odzivnih časov so sprejemljivi in kako jih izboljšati (npr. pravilna izbira formata slike, velikosti slik, poenostavljanje elementov DOM, vpliv elementov v spominu itn.). Za konec je pokazal še spletno aplikacijo, ki jo je razvil sam, in nas pozval, da jo preizkusimo (bit.ly/spotazombie, gl. Zombie Outbreak).

Slika 5: Utrinek s predavanja M. Mengella

DAMIR ARH: NOVOSTI V C# 6

Na predavanju smo spoznali novosti, ki jih prinaša MS Visual Studio (MS VS), predvsem novosti v sintaksi programskega jezika C#. Spremembe lahko razdelimo na štiri sklope:

- spremembe na nivoju izrazov,
- spremembe na nivoju sintakse,
- spremembe pri deklaracijah,
- spremembe pri uvozu statičnih članov tipov.

Spremembe na nivoju izrazov

Izraz NameOf

Občasno moramo podati ime določenega programskega elementa, npr. kadar:

- se sklicujemo na parameter, ki je npr. povzročil izjemo,
- poimenujemo ime spremenljivke ob programsko zagnanih dogodkih (npr. PropertyChanged event).

V takšnih primerih se v sporočilu o napaki sklicujemo na ime spremenljivke (gl. primer spodaj). Izrazi NameOf so v osnovi znakovni nizi; prevajalnik (angl. *compiler*) preveri, ali obstaja entiteta s takšnim imenom in programsko orodje MS VS ve, na kaj se navezuje.

```
(if x == null) throw new ArgumentNullException(nameof
(x));
```

Interpolacija nizov

Izraz String.Format in njegove izpeljanke so zelo uporabni za delo z nizi, čeprav so nekoliko nepregledni in obstaja možnost napak, predvsem kadar gre za oznake v zavitih oklepajih ({0}) v kontrolnem nizu z argumenti, ki morajo biti podani v točno določenem zaporedju.

```
var s = String.Format("Ime mi je {0}.", p.Name);
```

Interpolacija nizov omogoča, da se izraz lahko takoj postavi na pravo mesto v kontrolnem nizu z uporabo izrazov.

```
var s = $"Ime mi je {p.Name}.";
```

Vsebina je lahko kakršen koli izraz, tudi drug niz znakov.

Izraz za pogojne operacije z izrazom NULL

Pred ovrednotenjem elementa nekega objekta je treba preveriti, ali objekt obstaja in nima vrednosti *NULL*.

```
return arg != null ? arg.Length : 0;
```

Operator bo izvajal verigo dostopov in klicev metod samo, če je objekt različen od *NULL*.

```
return arg?.Length ?? 0;
```

Operator lahko uporabimo tudi za klice funkcij, ki so podane kot parameter v drugi funkciji ali kot samostojni dogodek (angl. *event*).

```
PropertyChanged?.Invoke(this, args);
```

Inicializacija indeksov

Inicializacija indeksov omogoča preglednejši način kreiranja slovarjev in prirejanja vrednosti nekega ključa.

```
var d = new Dictionary<string, string>
        {
           ["Name"] = "Damir"
        };
```

Inicializacija zbirk z razširitveno metodo Add

V C# 5 (gl. prvi primer) smo razširili metodo Add tako, da je prejela samo en argument in s tem napolnila vrednost slovarja.

```
var dictionary = new Dictionary<Type, string>();
dictionary.Add(typeof(string));
return dictionary;
```

V novi različici poteka inicializacija slovarja enako, vendar pa prevajalnik sam ugotovi, katero metodo mora poklicati, da bo slovar pravilno napolnjen.

```
var d = new Dictionary<Type, string>
        {
         typeof(string)
        };
```

Spremembe na nivoju sintakse

Sintaksa filtriranja izjem

Primer sintakse v C# 5:

```
try
{
throw new ArgumentException(null, paramName);
}
catch (ArgumentException exception)
{
if (exception.ParamName != "ignore")
        {
            throw;
        }
}
```

Kadar želimo obravnavati izjeme, ki so točno določenega tipa in vsebujejo točno določeno informacijo, uporabimo filtriranje izjem. Novost v sintaksi C# 6 je, da lahko že v sami deklaraciji stavka *Catch* povemo, katere izjeme so za nas zanimive.

```
try
{
throw new ArgumentException(null, paramName);
}
catch (ArgumentException exception) when (exception.
ParamName == "ignore")
{ }
```

*Izraz **Await** v blokih **Catch** in **Finally***

V C# 5 ni bilo dovoljeno uporabljati ključne besede *Await* v blokih *Catch* in *Finally*. Zato so programerji pisali različne kode in se tako izognili tej težavi. V novi verziji C# 6 tega problema več ni, saj je dovoljeno uporabljati ključno besedo *Await* v obeh omenjenih blokih.

```
try
{
await resource.OpenAsync(throwException);
}
catch (Exception exception)
{
await resource.LogAsync(exception);
}
finally
{
await resource.CloseAsync();
}
```

Spremembe pri deklaracijah

Samodejna inicializacija lastnosti

Ob inicializaciji se samodejno določi vrednost lastnosti. Inicializatorji se sprožijo v navedenem vrstnem redu skupaj z inicializatorji polj. Ker se to dogaja, preden je objekt v celoti inicializiran, se na to ne smemo nikjer sklicevati.

```
public string Name { get; set; } = "Damir";
```

Lastnosti samo z metodo Get

Samodejne lastnosti se lahko definirajo tudi brez metode Set. V ozadju se ustvari polje za branje (angl. *read-only*), ki se deklarira preko samodejnega inicializatorja ali konstruktorja.

```
public class GetterOnlyAutoPropertiesAfter
{
   public GetterOnlyAutoPropertiesAfter()
   {
      Surname = "Arh";
   }
   public string Name { get; } = "Damir";
   public string Surname { get; }
}
```

Uporaba izrazov namesto telesa funkcije (angl. expression-bodied function members)

Izrazi lambda se lahko deklarirajo kot konvencionalno telo funkcije ali kot izraz.

```
private readonly string _name = "Damir";
private readonly string _surname = "Arh";
private string FullName => _name + " " + _surname;
```

Spremembe pri uvozu

Uvoz statičnih članov tipov

Ta funkcionalnost omogoča, da se vsi statični člani nekega razreda uvozijo in so s tem na voljo brez dodatnega navajanja znotraj drugih razredov.

```
using static System.Console;
using static System.Math;
using static System.DayOfWeek;
class Program
{
   static void Main()
   {
      WriteLine(Sqrt(3*3 + 4*4));
      WriteLine(Friday - Monday);
   }
}
```

Predavatelj je poudaril, da so novosti v nekaterih pogledih dobre, v drugih manj, in da naj jih uporabljamo predvsem tam, kjer je to smiselno, saj bo morda nekoč kdo drug prevzel kodo in s tem tudi morebitne težave pri branju kode.

TINA MAZE, ANDREA MASSI: TEAM TO AMAZE

Na poslovnem srečanju sta bila gosta Tina Maze in Andrea Massi. V svojem prispevku sta poudarila, kako pomembno je, da se nemudoma in enostavno prilagodimo hitrim spremembam, ki se pojavljajo v športu, in kako delovati, kadar gre nekaj narobe, kadar pogoji niso idealni, kadar ni želenih rezultatov in še posebej, kadar ni denarja.

Andrea Massi je zelo pragmatična oseba. Vse, kar dela, dela z namenom, da bi čim prej dosegel svoj cilj ter da bi se izognil formalnostim in manjšim nevšečnostim. Končal je Fakulteto za šport v Rimu. S Tino je začel sodelovati leta 2002, ekipa Team to aMaze pa je bila ustanovljena leta 2008. V ekipo je najprej vnesel ekipni duh, potem enaka oblačila, ki povežejo ekipo in krepijo pripadnost ter povečajo samozavest, izboljšajo moralo in psihično povzdignejo tekmovalca. Tina je bila zelo dobra, še preden se je ekipi pridružil Massi, ki ji je s svojim pristopom pomagal do vrhunskih rezultatov. Marsikdaj so se še vedno pojavljale težave, vendar ne šteje neuspehov, temveč uspehe. Če bi se ukvarjal z neuspehi, potem bi pozabil, kakšen je cilj. Predavanje se je končalo z mislijo, da smo Slovenci preveč nesamozavestni in da se bojimo svoje majhnosti v družbi velikih. Če citiramo Massija: "*Nekje v družbi so mi rekli, da je to, kar delam s Tino, nesmiselno, da ste Slovenci majhni, pa sem jim v šali odvrnil, da je Lindsey Vonn (ZDA) prav tako velika kot vaša Tina, saj v višino obe merita 172 cm.*"

ANDY WIGLEY: THE WINDOWS 10 APP PLATFORM: AN INTRODUCTION TO THE UNIVERSAL APP PLATFORM

Andy Wigley se ukvarja z aplikacijami na platformi Windows za telefone, tablice in osebne računalnike. Microsoftu se je pridružil leta 2012, pred tem je bil pomemben član skupnosti razvijalcev mobilnih aplikacij. Deset let zaporedoma je dobil nagrado Microsoft Most Valuable Professional (MVP).

Predstavil je osnovne smernice operacijskega sistema (OS) Windows 10 skupaj z univerzalno aplikacijsko platformo (angl. *Universal App Platform* (UAP)). V nadaljevanju povzemamo članek z naslovom *Windows 10: an introduction to building Windows Apps for Windows 10 devices*, ki najbolje predstavlja njegovo predavanje na 20. Microsoftovi NT konferenci (Wigley in Nixon, 2015).

Dočakali smo enoten operacijski sistem Windows, ki deluje na vseh napravah Windows. Dobili smo enotno platformo operacijskega sistema; le-ta omogoča univerzalne gonilnike za strojno opremo in enotno platformo za

aplikacije ter zagotavlja univerzalnost aplikacij Windows, kar predstavlja pomemben razvojni dosežek.

Na nivoju operacijskega sistema pomeni to enotno, trajnostno in prilagodljivo osnovo, za razvijalce pa zagotavlja enotno in zanesljivo podlago API za naprave z OS Windows, in sicer od naprav interneta stvari (angl. *Internet of Things*), telefonov, igralne konzole Xbox, tablic do prenosnih in namiznih računalnikov in še veliko več (npr. Microsoft HoloLens – hologramski zasloni). Na sliki 6 je prikazan obljubljeni operacijski sistem Windows 10 z univerzalno aplikacijsko platformo (UAP), ki zagotavlja, da koda, ki jo napišemo enkrat, deluje povsod (angl. *write-once-run-everywhere*).

Slika 6: Univerzalna aplikacijska platforma omogoča aplikacije na vseh napravah družine Windows (Vir: Wigley in Nixon, 2015)

Pot k Windows 10

Leta 2011 je imel Microsoft tri platforme s tremi operacijskimi sistemi, na katere so želeli namestiti enoten Internet Explorer, kar je bila tehnično zelo zahtevna naloga.

Z Windows Phone 8 se je zgodil premik k združevanju enotne kode. Naslednji korak je bila uporaba istega jedra za Windows, Windows Phone in Xbox 360. Nato pa so se leta 2013, ko sta Xbox One in Windows 8 začela uporabljati isti operacijski sistem, že zelo približali enotnemu operacijskemu sistemu Windows, toda še vedno so vzdrževali tri različne operacijske sisteme.

Windows 10 je dobil priložnost združiti te tri različne operacijske sisteme. Družini naprav Windows so se pridružile še nove naprave, zato je postalo nujno, da je Windows 10 enoten operacijski sistem za vse naprave Windows, telefone Windows Phone in naprave Xbox pa tudi za vse prihodnje naprave in platforme Windows.

Microsoftu je uspelo. Windows 10 je majhno enotno jedro, ki deluje na vseh napravah družine Windows. Trdo inženirsko delo, izpeljano v tako kratkem času, se je obrestovalo. Windows 10 je enotna osnova za univerzalno aplikacijsko platformo (UAP), na podlagi katere bo mogoče napisati vse prihodnje programske aplikacije.

En operacijski sistem nima istih uporabniških vmesnikov za različne naprave. Platforma naprave in platforma aplikacije sta enaki, medtem ko so uporabniški vmesniki in funkcije jedra različni in prirejeni za pravilno uporabo različnih naprav.

Aplikacije Windows ne bodo usmerjene v OS Windows, temveč v aplikacijsko platformo – napravo. UAP je konsistenten aplikacijski model, znotraj katerega osnova API zagotavlja delovanje na vsaki napravi Windows.

Microsoft Office 2016 je zdaj del družine aplikacij UAP. Uporabljena je tehnologija XAML, z UAP pa so podprte tudi aplikacije DirectX in JavaScript (aplikacije Windows Web). Namizje Windows uvaja številne novosti, ki so narejene s tehnologijo XAML. Veliko pomembnih aplikacij OS Windows temelji na XAML.

Ni dovolj, da aplikacija deluje le na eni napravi. Dejanska vrednost se pokaže, ko lahko aplikacijo zaženemo na različnih napravah. Zaradi prilagodljivosti UAP lahko vključimo določeno kodo naprave v enotno binarno kodo, ki bo delovala na kateri koli napravi.

Imamo en Windows Store za vse aplikacije (telefone, tablice, namizne računalnike in Xbox), od koder je možno namestiti aplikacijo na ustrezno napravo. Windows Store tako zagotavlja, da je naša aplikacija ustrezno naložena na izbrano napravo.

Na posnetku predstavitve (A First Look at Building Windows 10 Universal Apps) je na voljo več informacij o univerzalni aplikacijski platformi Windows.

HALIS TABAKOVIĆ: POWERSHELL ZA VSAKOGAR!

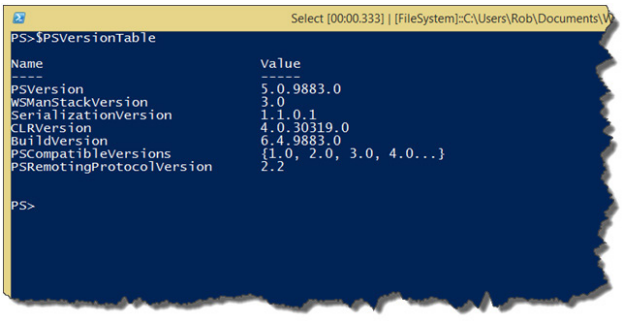
PowerShell je za sisteme Windows postal osnovna ukazna lupina in skriptni jezik v enem, ki sistemskim skrbnikom omogoča učinkovito upravljanje in avtomatizacijo opravil. Windows 10 in Windows Server 2016 imata privzeto nameščeno ukazno lupino PowerShell, V 5.0. Nova verzija ukazne lupine prinaša nekatere zanimive novosti, ki bodo povečale učinkovitost in izboljšale upravljanje sistemov Windows.

PowerShell, V 5.0, omogoča ustvarjanje razredov z uporabo sintakse, ki je značilna za druge objektne programske jezike. Novost je nov strukturirani informacijski podatkovni tok (angl. *structured information stream*), ki se lahko uporablja pri prenosu podatkov med skriptami in sistemskim okoljem.

DSC (Desired State Configuration) je lastnost, ki je v PowerShellu prisotna od verzije 4.0. Sistemskim skrbnikom omogoča enostavno upravljanje in konfiguriranje

strežnikov in delovnih postaj. DSC med drugim omogoča:

- enostavno nameščanje strežniških vlog in njihovih lastnosti,
- upravljanje nastavitev registra sistema,
- upravljanje datotek in map,
- upravljanje procesov in servisov,
- upravljanje lokalnih uporabniških računov in skupin,
- nameščanje programskih paketov tipa *.msi in *.exe,
- upravljanje spremenljivk sistemskega okolja,
- izvajanje skript PowerShell,
- samodejno konfiguriranje sistemov, ki odstopajo od načrtovane konfiguracije,
- enostavno poizvedbo konfiguracije na določenem sistemu.



Slika 7: Ukazna lupina verzije 5.0

Pogoj za uporabo DSC je, da morajo biti sistemi vsaj verzije Windows 2012R2, Windows 8.1 ali novejši. Podprti so tudi nekateri sistemi Linux. Za delovanje na sistemih Linux je treba namestiti Open Management Infrastructure (OMI). DSC lahko uporabljamo na sistemih v svojem podatkovnem centru in na sistemih, ki so v javnem oblaku. DSC predstavlja način, kako bomo lahko na enostaven način obvladovali tudi dokaj obsežno in raznoliko informacijsko okolje.

V verzijo 5.0 so dodani nekateri novi moduli in precej novih ukazov (cmdlets). Izpostavili smo nekaj zanimivejših. Modul PackageManagement omogoča poizvedbo in nameščanje programskih paketov z interneta. Nova lastnost, poznana pod imenom OneGet, je že vključena v Windows 10 Technical Preview. Uporabniki novih sistemov Windows bodo imeli možnost nameščanja programskih paketov z interneta, podobno kot to že precej časa počnejo na sistemih Linux. Dobrodošel je tudi modul Microsoft.PowerShell.Archive, ki uporabnikom omogoča enostavno kreiranje in upravljanje arhivskih (ZIP) datotek. Dodano je veliko število novih ukazov, npr. Clear-RecycleBin, Get-Clipboard, Set-Clipboard; njihov namen je predvsem avtomatizacija upravljanja večjega števila sistemov Windows.

Microsoft želi omogočiti podporo za upravljanje omrežnih stikal svojih izdelkov, kot je npr. Sistem Center Virtual Machine Manager. Nekatera omrežna stikala priznanih proizvajalcev (Cisco, Arista, Huawei) že imajo certifikat Windows Server Logo, kar jim zagotavlja interoperabilnost v ta namen. PowerShell, verzija 5.0, že vsebuje modul NetworkSwitch, ki bo omogočal enostavno upravljanje omrežnih stikal.

Reference

Trend Micro, 2015. [spletna stran] Dostopno na: http://www.trendmicro.com/us/index.html [27. 10. 2015].

Wigley, A. in Nixon, J., 2015. Windows 10: an introduction to building Windows Apps for Windows 10 devices. *MSDN Magazine*, [online] 30(5). Dostopno na: https://msdn.microsoft.com/sl-si/magazine/dn973012 [27. 10. 2015].

Drugi viri

How-to Geek, 2014. *Windows 10 includes a Linux-style package manager named "OneGet".* [online] Dostopno na: http://www.howtogeek.com/200334/windows-10-includes-a-linux-style-package-manager-named-oneget/ [27. 10. 2015].

Microsoft, 2015. *Get started with Windows PowerShell desired state configuration for Linux*. [online] Dostopno na: https://technet.microsoft.com/en-us/library/mt126211.aspx [27. 10. 2015].

Microsoft, 2015. *What's new in Windows PowerShell.* [online] Dostopno na: https://technet.microsoft.com/en-us/library/hh857339.aspx#BKMK_new50 [27. 10. 2015].

Microsoft, 2015. *Windows PowerShell desired state configuration overview*. [online] Dostopno na: https://technet.microsoft.com/en-us/library/dn249912.aspx [27. 10. 2015].

Rutkas, C., 2015. *A First Look at Building Windows 10 Universal Apps*. [online]. Dostopno na: https://channel9.msdn.com/Shows/Inside-Windows-Platform/A-First-Look-at-Building-Windows-10-Universal-Applications [27. 10. 2015].

Snover, J., 2014. Windows management framework V5 preview. *Windows Server Blog*, [blog] 3. april. Dostopno na: http://blogs.technet.com/b/windowsserver/archive/2014/04/03/windows-management-framework-v5-preview.aspx [27. 10. 2015].

Marko Belej, Luka Juršnik, Marko Kabaj, Andrej Korošec, Andrej Šerod, Gorazd Taciga, Janko Žigart