
nie
NIE
NIE
nie
nie
NIE
nie
NIE
nie
nie
Nie
NIE
nie
nie
nie
nie
NIE
Nie
Nie
nie
nie
nie
nie
nie
nie
nie
K.P. 11.10.13

# Adam Cebula "O sztuce miotania klątw, czyli dużo przykładów i mało refleksji w trybie tekstowym"

*Fahrenheit Crew*

Cóż, muszę uprzedzić: ten tekst może przynieść straty. Jeśli nie chcesz ich ponieść, nie czytaj; jeśli chcesz się dowiedzieć, jakie to straty i dlaczego, nieuchronnie doznasz tych strat, zaryzykuj. Więc jak to szło? *Lasciate ogni speranza voi ch'entrate.*

Jak powszechnie wiadomo - to młodzi ludzie są za pan brat z techniką, a takie dinozaury, jak ja, to najwyżej mogą biegać do nich po rady. Tylko dlaczego mam wrażenie, że jest inaczej? A miewam. To dość odległa od codzienności dziedzina, ale wystarczy poprosić delikwenta, by zrobił wykres z krzywą dopasowaną do punktów, a nie przebiegającą przez nie, i już się okazuje, że młody wilczek wylądował na Dzikich Polach.

Pamiętam ciężkie zdziwienie młodego człowieka, że używamy jeszcze dyskietek. A używamy, dlaczego miałbym zaniechać? Kilka razy pisałem, że póki działają stacje dyskietek, to całkiem skuteczny sposób na tworzenie kopii bezpieczeństwa. Oraz że wbrew dość rozpowszechnionym opiniom skuteczność takiej kopii jest ogromna. Prawdopodobieństwo uszkodzenia dyskietki nawet w przypadku awarii maksymalnej komputera, czyli wystawienia przez zasilacz za wysokiego napięcia na wszystkich wyjściach, jest nikłe. Można je wyzerować poprzez wysunięcie nośnika ze stacji. Kopia zapasowa utworzona na osobnej partycji twardego dysku, zewnętrznego nośnika podpiętego przez USB, owszem - ma sens, ale nie zabezpiecza już nawet przed sprzętową katastrofą, ale przed awariami informatycznymi, z których najczęściej zdarzają się pomyłki użytkownika. I te są najgroźniejsze.

Szok kulturowy, jaki przeżył młody człowiek, wiąże się jeszcze z brakiem świadomości, jaka pojemność nośników jest nam potrzebna. Ależ oczywiście, jeśli tworzymy prezentację multimedialną, to nie zachowamy jej na dyskietce. Ale już stronę WWW, pisaną w prostym HTML-u z atrakcjami muzycznymi w postaci plików midi, możemy.

Przekonanie, że dyskietka jest do niczego, bierze się chyba stąd, że na co dzień używa się aplikacji tworzących ogromne pliki. Są takie, które produkują małe, ale to umyka uwadze. Myślę, że ważniejszą obserwacją jest intuicyjne przekonanie o nieprzydatności czegoś nie podparte wiedzą, a jedynie tylko tym, że to "archaiczne". Pojemność dyskietki odpowiada mniej więcej 500-700 stronom maszynopisu, jeśli posługujemy się formatem txt. Można więc spokojnie zmieścić na niej sporą powieść.  Jest wszakże jeden problem: na przykład w dystrybucjach opartych na Debianie ktoś coś sknocił i poprzez zwykłe kliknięcie dyskietka się nie... ekhem... montuje. Właśnie: nie chodzi o to, aby pokazała się ikonka czy okno, ale by nastąpił proces zamontowania napędu w systemie. Magia. Trzeba na przykład wydać zaklęcie: *udisks --mount /dev/fd0*.

To nieprawda, że z młodymi jest kiepsko, a starzy to ho, ho. Nie: coś się robi z tak zwaną - kiedyś, dziś termin spadł z plakatu - kulturą informatyczną. Do komputera podchodzi się nie z głową, ale z kobiecą intuicją. Intuicja reaguje na kolory, kształty, oraz po sroczemu, na efekt nowości. Jak coś tej intuicji nie połechce, to ta wysyła sygnał, że się nie nadaje to niełechcące. Na tę kobiecą intuicję starają się oddziaływać i programiści, i na przykład recenzenci oprogramowania (bardziej w cudzysłowie, niż rzeczywiści), i dziennikarze komputerowi (raczej w cudzysłowie). Najważniejsze, by było nowe. Postęp w informatyce jest bożyszczem. Jak coś jest stare, to wystarczy, by do niczego się

nie nadawało.

Nawet sobie nie zdajemy sprawy, że za takie podejście płacimy. I pieniędzmi, i bezradnością. Pomijam tu spotkanie z oczami dinozaura pełnymi wyrozumiałego zrozumienia. Postawa taka kosztuje czas, często niewykonane zadania, opinię w pracy i pomiędzy znajomymi, a czasami dochodzi do, jak się zdaje, prawdziwych dramatów. "Sowicie wynagrodzę za pomoc w odzyskaniu danych". "Na pendrive'ie była moja praca magisterska", czyta człowiek i w głowę zachodzi, jak mogła istnieć tylko jedna kopia ważnych danych? Sieć jest pełna postów bez odpowiedzi, coś nie konwertuje, coś chodzi strasznie długo... No i jak pogrzebiemy w archiwach, to właściwie problemy są takie same od lat, choć w komputerach - co z zeszłego roku, to już stare.

Niestety, postęp w informatyce to przede wszystkim algorytmy. Widzimy właściwie tylko interfejs, ale to, co może zrobić oprogramowanie, zależy nawet nie od szybkości procesora, od wielkości pamięci operacyjnej, ale od tego, czy mamy pomysł, jak problem rozwiązać. Jak jest algorytm, to można kombinować: jakie zasoby, jaka ilość pamięci, jak szybki procesor, ale zwykle nie mamy zielonego pojęcia "jak to ugryźć". Zazwyczaj nowy algorytm  przewraca funkcjonalność oprogramowania.

Otóż, o ile bardzo szybko zmieniało się w ostatnich latach "opakowanie" aplikacji, to z nowymi algorytmami jest raczej tak sobie. Mam wrażenie, że współczesny tryb kształcenia informatyków odpuścił sobie to, co stanowi istotę, czyli właśnie wynajdowanie sposobów rozwiązywania zadań. Bo to wymaga niestety bardzo dobrej znajomości i bebechów komputera, i po prostu matematyki.

To chyba zasadnicza przyczyna tego, że funkcjonalność oprogramowania czasem bardzo starego i bardzo nowego jest bardzo podobna. Funkcjonalność oprogramowania, które w istocie jest nowe, ale siedzi w starym opakowaniu, jest oczywiście "nowa". Sęk w tym, że to opakowanie determinuje spostrzeganie aplikacji. No i mamy taki efekt, że posługiwanie się programami pracującymi w trybie tekstowym, albo "jakoś inaczej wyglądającymi" jest często nie do przeskoczenia dla tak zwanego młodego zdolnego informatyka. Albo i starego, który wiedział i zapomniał.  Wiele razy o tym się przekonałem, próbując jakiegoś nieszczęśnika naciągnąć na zapoznanie się z opisanym już dawno **gnuplotem.** I co z tego, że jest jakiś na poły mityczny program, który coś robi dobrze - podobno - skoro na oko widać, że użyć go się nie da?

Konsekwencją intuicyjnego podejścia do komputera jest bezradność. Chcesz coś zrobić, ale masz w teamie kogoś, kto nie zrobi. Konsekwencją jest często wypisywanie w różnych na pół informatycznych portalach bzdur, zwykle sprowadzających się do naciągania na kupienie czegoś, co mogłoby rozwiązać opisywany problem, gdyby działało. Tymczasem zwykle mamy gotowe narzędzia, nie mamy problemu, a tylko dziury w wiedzy. Mamy przekonanie, że powyżej pewnego stopnia komplikacji to już albo nie wolno, albo z pewnością nie damy rady.

Powiem od razu, że ostatni Windows z jakim miałem do czynienia to XP. To o czym piszę, niestety, działa  pod Linuksem. Celem nie jest detaliczne opisanie aplikacji, ale ukazanie ciut szerszych okoliczności. Otóż: z tego powodu, że aplikacje które nie otwierają swoich okien "nie istnieją" wiele osób powtarza opinię, że pod Linuksem oprogramowania nie ma. Jak przychodzi zrobić coś konkretnego, to musimy wrócić do Windows. Bo nie widać okien.

Oprogramowanie uniksowe działa w trybie tekstowym. Dzięki temu potrafi zadziałać o wiele szybciej przy zapotrzebowaniu na mniejsze zasoby. Zwykle. Bywa i odwrotnie. Proponuję wywołać konsolę tekstową i wypróbować polecenie - na przykład ***find***. Jak sama nazwa wskazuje, służy ono do poszukiwania w czeluściach maszyny. Proszę o chwilę cierpliwości, lubię dygresje. Polecenie w rzeczywistości jest programem o całkiem sporych możliwościach.  Możesz coś się o nich dowiedzieć wpisując polecenie ***find --help***. Warto zapamiętać tę sztuczkę. Bardzo wiele poleceń pracujących w

trybie tekstowym zareaguje wypisaniem podstawowych wskazówek. Niestety, zazwyczaj dla nowicjusza zupełnie niezrozumiałych. Ale jest kolejna sztuczka: ***man find***. W rzeczywistości wywołujemy program man, system manuali dla Linuksa, który wyświetla akurat to, co dotyczy polecenia find.

W trybie tekstowym da się przeżyć dzięki temu, że zachowywane są pewne standardy. Jednym z nich jest, że nie musisz wiedzieć wszystkiego. Wiedza ma hierarchiczną organizację, są ogólne zasady, które wystarczy poznać i kierować się nimi przy rozwiązywaniu szczegółowych problemów. To system przystosowany też do tego, że coś tam w głowie zaczyna świtać i możesz sobie już pomóc, jeśli znasz tylko odpowiednie chwyty. Na przykład w głowie się kołacze, że to było coś na fi. Wpisujemy ***fi*** i wciskamy klawisz tab. Wówczas komputer wypisze wszystkie polecenia zaczynające się na fi. No i może się zdarzyć, że już jesteśmy w domu. Może też być tych poleceń za wiele, ale często ten myk działa. Tak czy owak, pamiętać warto o tab, natomiast polecenia możemy tak mniej więcej.

Podobnie uniwersalne jest polecenie man. Chcemy sprawdzić, czy mamy dokumentację do jakiegoś polecenia - wpisujemy ***man polecenie***. Mamy też inne podręczniki, np. info, i wywołujemy pomoc taką samą składnią: ***info polecenie***.

Niestety, manuale są pisane z myślą o takich, co już sporo wiedzą, a jedynie chcą sobie przypomnieć, co i jak. Taka przykrość, ale to wynika z tych okoliczności, że albo zrobimy skuteczną zwięzłą pomoc dla kogoś, kto już z programem pracował, albo rozlazły podręcznik z przykładami dla nowicjusza. Początki bywają trudne.

Mówiliśmy chyba o poleceniu find? Jak go użyć? Do znalezienia pliku **find -name '*nazwa_pliku*'**. Zwykle tak tego używam. Otóż teraz znów dygresja i coś ogólniejszego. W trybie tekstowym jest całkiem sporo skutecznych ułatwień. Dinozaury zechcą wybaczyć, że napiszę o czymś zupełnie oczywistym. To znak "*". Gwiazdka zastępuje ciąg znaków. W miejscu gwiazdki w poszukiwanym wyrażeniu może wystąpić dowolny ciąg znaków. To działa w większości poleceń (nie pamiętam, gdzie nie działa...) i pozwala na szukanie czegoś, o czymś mamy trochę zielone pojęcie. Na przykład plik, który pewnie się nazywał fajne_zdjęcie, ale nie wiemy w jakim jest formacie czy to png czy jpg a może JPG? W Linuksie niestety duże i małe litery są rozróżniane. Jak nie wiemy, piszemy ***find -name 'fajne_zdjecie.***'  Można napisać '***kawałek_nazwy***' i wówczas zostanie wypisane wszystko co ma ten "kawałek_nazwy" w sobie, ale różne początki i końce.  Jeszcze jedno: nie chcesz mieć kłopotów - nie używaj w nazwach plików polskich znaków. Okazji, by coś się przez to skaszaniło, jest coraz mniej , ale o ile nie ma jakiegoś ważniejszego imperatywu to lepiej mieć w nazwach plików i katalogów pisownię polskawą. Bo może się okazać, że np. program find odczyta ą czy ę jako zestaw "backslaszy" z gwiazdkami i pytajnikami.

Jeśli chcemy zobaczyć listę wszystkich plików JPG, napiszemy ***find -name '*.jpg'***. Zapewne przeleci nam przez ekran długaśna lista, dziś każdy ma mnóstwo zdjęć na dyskach. Trochę to bezużyteczne, jeśli pracujemy w trybie tekstowym. W graficznym zwykle okno terminala ma suwak i możemy wrócić do początku tekstu, ale nie w "czystym tekście". Możemy natomiast zrobić coś takiego: ***find -name '*.jpg'> lista_fotek.txt.***

To sztuczka zwana przekierowaniem. Rzecz wymagałaby szerszego omówienia, ale w uproszczeniu programy zwykle kierują wynik na standardowe wyjście i jest nim po prostu terminal. To „standardowe wyjście” to termin który dla zrozumienia  potrzebuje trochę gadania, ale przyjmijmy, że terminal i że znakiem > zmuszamy program do zapisania wyników w pliku o nazwie następującej po nim. Jeśli użyjemy znaku >> to następuje dopisanie wyników do końca pliku, który może już istnieć. Jeśli pliku nie ma, to w obu przypadkach powinien on zostać utworzony.

Polecenie ***df*** wypisze nam listę partycji czy urządzeń do zapisywania danych np. wymiennych nośników. Jeśli zabieramy się do zrujnowania systemu, a zamierzamy "zaorać" jedynie partycję systemową, to poleceniem ***df>moje_partycje.txt*** zapiszemy sobie konfigurację komputera.

Kilka razy o tym już pisałem, ale przypomnę, bo było to dawno. W Linuksie sensowna konfiguracja wygląda tak, że katalog domowy mamy na osobnej partycji. To nam pozwala uniknąć po wykoszeniu starego systemu ponownej konfiguracji poczty, lwiej części programów, typu przeglądarki internetowe czy programy do ftp. Bo zakładają one w naszym katalogu domowym swoje pliki i katalogi konfiguracyjne i nawet gdy wymienimy dystrybucję Linuksa na "zupełnie inną" to powinno być tak, że np kmail program do obsługi poczty "wstanie" ze wszystkimi odebranymi listami, kontami pocztowymi, oraz spamem w katalogu spam. Przeglądarka internetowa zachowa swoje zakładki, także program typu gFTP do przesyłania w protokole ftp.

Dzięki temu reinstalacja systemu w Linuksie może odbyć się dość gładko. Podobno to już możliwe w Windows?  No i to jest powód, by wykonać np. zrzut stanu konfiguracji.

Wracając do find: a jeśli mamy ochotę tylko obejrzeć listę naszych plików? Nie mamy ochoty tworzyć kolejnego pliku zaśmiecającego system? Piszemy ***find -'name *.jpg'|more.***

W systemie mamy polecenie **more**, które służy do czytania plików txt i działa tak, że wyświetla jeden ekran tekstu. Gdy naciśniemy "enter", tekst zaczyna się przewijać.  Dzięki temu nie ucieknie nam poza ekran. Proste czytanie pliku wywołujemy po prostu poleceniem ***more nazwa_pliku.txt***. Ale sens more widać w takich konstrukcjach, jak wyżej. Mamy tam jeszcze jedno dziwo - pionową kreskę. W uproszczeniu działa ona tak, że kieruje wynik działania jednego programu do drugiego, znajdującego się za kreską. Oczywiście musimy wiedzieć, czy te programy się nie pogryzą, i nie zawsze to zadziała. Tym niemniej sztuka jest powszechnie stosowana dla uproszczenia sobie życia. Przykład: chcesz przeglądnąć zawartość katalogu z mnóstwem plików? Piszemy: ***ls |more***. Gdyby kto nie wiedział, **ls** to polecenie wyświetlenia zawartości katalogu.

Polecenie **find** ma mnóstwo możliwości: możemy znajdować pliki katalogi, pliki o określonej wielkości, czasie utworzenia, linki twarde i symboliczne (taka osobliwość, ponoć już jest w Win8). Można wykonywać inne działania na znalezionych obiektach. Uruchamiamy to za pomocą **exec.** ***find . -name '*.jpg' -exec rm {} \;*** powinno wyciąć wszystkie jpegi z katalogu w którym polecenie zostało uruchomione. Ciąg znaków **-exec polecenie {} \;** uruchamia wykonanie operacji na obiekcie, w tym wypadku usuń, czyli **rm**. Kropka pomiędzy **find** i **-name** mówi, że chodzi tylko o ten katalog. Nie wypróbowywałem polecenia, więc jeśli to robisz, to na swoje ryzyko.

W Windows było całkiem sporo bardzo specjalnych programików do czyszczenia systemu ze śmieci. Jak w Linuksie znaleźć na przykład zduplikowane pliki? Za pomocą programu  fdupes.  Program wypisuje na konsole listę zduplikowanych (i rozmnożonych więcej razy) plików oddzielając kolejne serie pustymi liniami. Jak to uruchomić? ***fdupes -R /home/katalog_domowy>>zduplikowane.txt*** i otrzymamy plik z listą z całego naszego katalogu, w którym rozpoczęliśmy poszukiwania.

Z nieznanych mi powodów dość często użytkownicy mają przekonanie, że pod Linuksem brakuje narzędzi do odzyskiwania danych, albo że jeśli są, to takie nie dla ludzi. Jest ich całkiem sporo. Pominę tu takie, które uruchamiają się same i np. dokonują automatycznej defragmentacji dysków. Zanim jednak przejdziemy do konkretu, trochę teorii na temat teorii.

Istnieje coś takiego jak "czysta nauka" oraz "nauki stosowane". Stosowane, wiadomo, stosujemy, ale czysta wiedza cieszy się ostatnio złą sławą. To synonim gadania po próżnicy, tracenia czasu na nikomu nie potrzebne poznawanie wymysłów, które służą jedynie temu, by zaprezentować się publiczności jako człowiek niesłychanie wykształcony, a nawet jeśli nie, to przynajmniej

wtajemniczony. Wiedza ogólna jest uznawana za niepotrzebną. Właśnie się staram wykazać, że tak nie jest. Wiedza ogólna pozwala nam unikać zapamiętywania szczegółowych reguł tyczących każdego przypadku z osobna. Np. taką wiedzą ogólną jest to przekierowanie uruchamiane znaczkiem ">". O ile jakiś program kieruje wynik swego działania na ekran w postaci znaków alfanumerycznych, możemy z niego zrobić plik.

O ile wiemy, jak działają nośniki danych, to zrozumiemy bez kłopotu, co robią programy. które na nich operują, i zastosujemy te same operacje w różnych okolicznościach i z różnymi aplikacjami. Wiedza ogólna może być bardzo mętna, by wystarczyła do połapania się, o co tu chodzi. Przykład: nośniki pamięci prawie zawsze działają tak, że mają rodzaj katalogu (tablica alokacji plików), w którym jest opisane gdzie co jest przechowywane, i tzw. obszar danych. Katalog jest zwykle malutki, obszar danych duży.

To, jak jest zorganizowany nasz katalog, w uproszczeniu nazywamy systemem plików. Systemy plików mają swoje ciekawostki istotne dla usera: np. nie pozwalają zapisywać plików większych niż 4 GB. To nas spotyka na kartach pamięci czy popularnych "gwizdkach". System plików jest powodem tego, że rozróżniamy operację kopiowania i tworzenia obrazu nośnika. Kopiowanie odbywa się tak, że komputer czyta katalog i na podstawie zawartych w nim informacji znajduje dane potrzebne do zapisania na nowym nośniku poszczególnych plików. W tworzeniu obrazu katalog jest ignorowany. Komputer opisuje zawartość poszczególnych tzw. sektorów czy bloków. Większość nośników jest tak zorganizowanych, że ma rodzaj szufladek, które możemy zaadresować i wydobyć ich zawartość. Szufladką na dysku jest sektor, fragment ścieżki magnetycznej, czy kawałek płyty CD.

Jeśli obsługujemy nośnik "normalnie", to komputer wyczytuje z katalogu adresy sektorów, a z sektorów dopiero dane. Jeśli kasujemy jakiś plik, to nie ma sensu zamazywać zawartości sektorów, ale w rozpisce zawartości dysku są one zaznaczane jako do zapisania. Dane tam dalej są i to daje nadzieję na odzysk danych. Warto też sobie uświadomić, że o ile zostanie uszkodzony obszar danych to najwyżej utracimy jeden plik. Jeśli na dysku padnie MBR - Master Boot Record, w który mamy rozpiskę partycji, to jest katastrofa.

Mówiąc w dużym uproszczeniu, to z powodu istnienia systemów plików nie da się utworzyć dysku startowego przez kopiowanie, ponieważ katalogi opisujące zawartość dla dysku systemowego muszą zawierać informację, że to z tego dysku komputer ma wystartować, a komenda kopiowania nie operuje na owych katalogach. Dlatego gdy tworzono dyskietki startowe robiono to mniej więcej tak: ***dd if=obraz_dyskietki.img of=/dev/fd0*** (przykład za Wikipedią).

Kluczowym programem dla powodzenia operacji jest **dd**. Akronim dd jest tłumaczony jako data definition, ale też destroy disk. Mniejsza o te akronimy. Ten program może dokonać rzutu tego tzw. obrazu nośnika na inny nośnik.

Tu kolejna ogólna dygresja. Gdy uruchamiasz polecenie, masz do rozwiązania problem jego składni. To NIE są zaklęcia, to ma sens i porządek. Prawie wszystkie polecenia w Linuksie wyglądają tak: ***polecenie -opcja co_przetworzyć***. Nasz dd ma inną składnię. Nie musisz pamiętać składni wszystkich poleceń, nie musisz pamiętać nawet jaka jest norma dla Linuksa, ale trzeba zapamiętać, że w pomocy do polecenia szukamy składni. Jak znajdziemy składnię, szukamy znaczenia opcji, to co przetworzyć jest zwane zwykle argumentem i to mamy, bo to zapewne stało się początkiem naszych tarapatów. Zwykle zadanie wygląda tak: jak poinformować co przetworzyć, znaleźć magiczne literki, albo słowa zwane czasem przełącznikami, które mówią jak przetworzyć i wskazać gdzie złożyć wynik. Składnia dd wygląda tak że co kopiować - piszemy **if=zrodlo_danych**, do czego - **of=plik_docelowy opcja=coś.**

Program dd może służyć na kilka różnych sposobów do ratowania danych. Możemy np. za jego

pomocą zrobić kopię MBR (czyli katalogu gdzie co jest) na nośniku. Bo walnięcie się obszaru danych jest zwykle mniej bolesne. Nasz dd może też odczytać np. uszkodzoną partycję (http://www.jakubiak.eu) w taki sposób: ***dd if=to_co_popsute of=wolne_miejsce_na_dysku conv=notrunc, noerror, sync bs=4096***. Opcja noerror każe czytać pomimo błędów, bs to wielkość bloku danych dla nośnika, razem z dwiema pozostałymi opcjami polecenie będzie się starało zbudować czytalny obraz nośnika - czego niestety nie sprawdziłem do końca - z uszkodzonymi miejscami zastąpionymi zerami.

Tu zrobię przeskok w wątku odzyskiwania plików do programu z grubsza do niczego. Polecenie zwie się tar. Od Tape Archiwer. Operacja łączy kilka(set) - kilka tysięcy plików w jeden. To zwykle część operacji przed spakowaniem danych, popularnie zwanej operacji "zipowaniem". Samodzielne używanie polecenia tar wydaje się bez sensu, ale w przynajmniej jednym wypadku to może być operacja, która umożliwi powodzenie całej akcji: podczas przesyłania plików za pomocą protokołu ftp.

Na wszelki wypadek powiem, że to osobna metoda na przenoszenie danych miedzy dwoma odległymi maszynami poprzez sieć, która służy do transferu dużej ilości danych. Są one przesyłane właściwie z dysku do dysku. Dlatego nic po drodze nie zapcha się, nie ma maksymalnej wielkości załącznika i innym temu podobnych niespodzianek.

Programy do obsługi protokołu ftp potrafią słać całe katalogi, ale niestety operacja składa się z automatycznego przesyłania pojedynczych plików. Tylko w nakładce graficznej na program widzimy ją jako jeden proces. W sytuacji, gdy łącze jest szybkie, nie ma co sobie głowy zawracać, ale gdy połączenie się rwie, zobaczymy, że program zatrzymuje się na pojedynczych plikach, usiłuje zakończyć transfer, nawiązać nowe połączenie i tak dalej. Wykonanie "tar" na danych może nam znakomicie przyspieszyć operację. Ceną za to jest konieczność "roztarowania" po odebraniu danych. Ale o ile odległa maszyna nam pozwala na zalogowanie, możemy to zrobić sami. Bardzo często zrobi to odbiorca. Tarowanie jest proste... No w zasadzie. Robimy to poleceniem ***tar -cf nazwa_spakowanego_pliku.tar to_co_jest_do_spakowania***. Można inaczej: ***tar --create --file zarchiwizowane.tar katalog_do_archiwizacji***. Jak się dobrać do archiwum? ***tar --extract --file nazwa_archiwum.tar.***

Można sobie wyobrazić taką sytuację: mamy 3 karty pamięci o rozmiarze 4 GB i 4 pliki po 2,5 GB. Co zrobić? ***tar --create --tape-length 4G -f pierwszy_kawalek_archiwum.tar katalog_z_wrednymi_plikami.*** Czyli robi z plików jeden i następnie dzielimy go na kawałki po 4 GB.

Program uruchamia się trybie interaktywnym, czyli wymaga obecności operatora. Produkuje kawałek danych o zadanej wielkości, po czym prosi nas o przygotowanie nowego nośnika. W tym momencie wpisujemy z klawiatury literkę **n** i podajemy nową nazwę pliku najlepiej dodając kolejną liczbę. Tak programy do zipowania w Windows robiły to już dawno, ale tar... jeszcze dawniej. Dlatego mamy tę przykrość, że trzeba podać rozmiar nośnika. Jeśli nie chcemy kombinować, bo da się to (wykrycie rozmiaru) zrobić automatycznie, ale chyba szkoda czasu.

Do wydobycia danych z naszych kawałków użyjemy komendy: ***tar --extract --multi-volume -f pierwszy_kawalek_archiwum.tar***

W Linuksie do dzielenia plików częściej używa się split razem z tar. Jakoś tak: ***tar c katalog_z_danymi | split -d -b 1024MB - moje_kawalki.tar***

Przełącznik (opcja) **-d** każe numerować kolejne kawałki archiwum. Opcja **-b** poprzedza podanie rozmiaru kawałka, w tym wypadku 1024 MB. Połączenia kawałków dokonujemy poleceniem: ***cat***

***moje_kawałki.tar*.cale.tar*** przy czym gwiazdka zastępuje kolejne numery nadawane uprzednio przez split.

Scalanie wielu plików w całość poprzedza zwykle operację kompresji danych. Współcześnie, mam wrażenie, ona "spadła z afisza", dawniej nośniki były relatywnie drogie, a dane w pamięciożernych formatach. Dziś są one zwykle z urodzenia spakowane i ponowna ich kompresja niewiele daje. Warto jednak wiedzieć, że w Linuksie mamy mnóstwo programów, które działają w oparciu o różne algorytmy. Dają one różne rezultaty. Chyba gwiazdą ostatnich lat jest 7-zip. Otóż nie będę się specjalnie tym zajmował, możemy oczywiście połączyć operację łączenia plików z operacją kompresji pisząc: ***tar czf*** itd. Dodaliśmy literkę **z**, która oznacza wybrany w tym wypadku, nazywany bzip, algorytm kompresji.

Jest rzadko używany typ kompresji, który pozwoli nam zamknąć dygresję i powrócić do wątku odzyskiwania danych: **lzip**. Jest on polecany dla długiego przechowywania danych i podobno dobrze współpracuje z mocnym programem do ratowania straconych nośników **ddrescue**. O ile o algorytmie lzip nie mogę wiele powiedzieć, to **ddrescue** z pewnością należy do jednych z najmocniejszych narzędzi informatycznych. Pracuje w trybie tekstowym. Piszemy na przykład: ***ddrescue -n -b2048 /dev/cdrom cdimage logfile*. Pójdzie też *ddrescue /dev/dvd ścieżka_do_obrazu.iso ścieżka do pliku log***. Warto zwrócić uwagę: plik log, w którym zapisany jest postęp pracy, jest potrzebny, nawet bardzo. Przełącznik **-n** poleca się ze względu na to, że wyłącza część operacji, na których program może powisieć i kilka dni. Nie wiem, czy kiedyś rezygnuje, na uszkodzonych płytach wyłączam go za pomocą kombinacji klawiszy ctrl+c. To warto też zapamiętać, działa w przypadku większości programów uruchomionych w konsoli. Warto też zapamiętać, że nie musimy wszystkiego wiedzieć, wystarczy, że wiemy jak sobie poradzić.

Dla przypomnienia: program odzyskuje OBRAZ nośnika, nie poszczególne dane.  Oznacza to, że po zakończonym sukcesem odzysku dostaniemy np. plik iso obrazu dvd, który musimy, jeśli chcemy dostać się do jego zawartości, zamontować w systemie. Trzeba wklepać takie coś: ***mount -t iso9660 -o loop,ro mój_obraz.iso***

Niestety, trzeba się napukać,  ale też na obrazie możemy uruchomić program, który naprawia błędy informatyczne. Nie poszedł mi program photorec również doskonały, nie zobaczył obrazu płyty, ale owszem udało się uruchomić dvdisaster. A, oczywiście uruchomienie photorec to kwestia pokombinowania, w najgorszym razie będziemy musieli wypalić obraz uszkodzonej płyty.

Program **ddrescue** ma ciekawą możliwość: mianowicie, jeśli mamy dwie takie same kopie uszkodzonych danych, to program po wykonaniu obrazu z jednej kopii,  na podstawie pliku log, poszuka w drugim nośniku tylko uszkodzonych fragmentów. Prawdopodobieństwo, że np. na dwóch płytach porobiły się nam dziury w tych samych miejscach, jest niezwykle małe - w rezultacie mając dwie kopie odzyskamy prawie na pewno swoje dane. Miałem dwie płyty, na jednej ponad 550 błędów, na drugiej 390  i kilka, i po operacji ratowania znalazłem jeden uszkodzony plik.

Kilka słów o **dvdisaster**. Poniekąd to nie z tej bajki aplikacja, bo posiada ładnego gui-a, ale nie do końca jest to program typu aplikacji dla normalnych ludzi. Albowiem normalny człowiek nie musi zawracać sobie głowy tym, co to jest na przykład obraz płyty, i rozróżniać go od jej zawartości. Otóż programu możemy używać do sprawdzania płyt i to jest operacja dla tych normalnych ludzi. Po prostu wsadzamy płytę do czytnika, odpalamy program i w oknie wynajdujemy przycisk scan. Otworzy się okno z obrazkiem płyty i wyświetlaniem postępu operacji. Jeśli na płycie znajdą się dziury, w danych zostaną zaznaczone na czerwono. To już całkiem sporo. Mamy jeszcze wykres prędkości odczytu. Jeśli nie jest gładką wznoszącą się krzywą wiemy, że płyta nie jest w dobrym stanie i należy ją ponownie nagrać.

Możemy jednak wykonać plik danych korekcyjnych, które w razie katastrofy służą do odzyskiwania danych. Tu już nie jest łatwo, albowiem **dvdisaster** wymaga rzeczy niedopuszczalnej w np. komercyjnym oprogramowaniu: użycia zewnętrznej aplikacji. Na przykład **k3b**, za pomocą którego nagramy najpierw obraz ISO danych. Następnie wskazujemy naszemu programikowi ten obraz i każemy wykonać plik danych korekcyjnych. Mamy dwie możliwości: dołączamy ów plik do obrazu ISO, albo zapisujemy na osobnym nośniku. Drugie jest mądrzejsze, jeśli chodzi o bezpieczeństwo danych, ale o ile nie uwzględnimy okoliczności zagubienia owych korekcyjnych plików. A, to raczej...

Trzeba wybrać metodę augmented (augmenting) w ustawieniach programu. W takim wypadku musimy sprytnie wykonać obraz ISO, który będzie ok. 20% mniejszy od pojemności płyty. Tyle zaleca twórca programu. Nasz **dvdisaster** przerobi ów obraz, dodając dane korekcyjne. Będą one niewidoczne z poziomu normalnego czytnika. Więc wskazujemy obraz, wciskamy create i po dość długim oczekiwaniu mamy zabezpieczony obraz, który możemy wypalić.

Jeśli płyta zacznie się sypać, musimy uruchomić **dvdisaster**, który z pomocą danych korekcyjnych odtworzy nam obraz płyty. Co sprawdziłem, możemy najpierw uruchomić **ddrescue**, za jego pomocą utworzyć obraz płyty z danymi korekcyjnymi i po jego zamontowaniu w systemie ten obraz potraktować naszą aplikacją. Warto zauważyć, że nagranie drugiej kopii płyty na sypiącym się nośniku, typowo R-W, zwykle one ulegają po kilku latach destrukcji, pomimo tego, że druga kopia będzie mocno dziurawa, może znacząco podnieść bezpieczeństwo danych. W razie czego za pomocą **ddrescue** reperujemy kopie i możemy się spodziewać, że każdy sektor odzyskany albo w obszarze danych korekcyjnych, albo zwykłych danych pozwoli nam naprawić błędy. Oczywiście, im lepsze nośniki - tym bezpieczniej, ale warto zauważyć, że dzięki narzędziom do odzyskiwania danych celowe jest użycie nawet bardzo kiepskich, właśnie jako dodatkowej kopii, z której coś się odczyta.

O skuteczności **dvdisaster** wiele nie powiem, nie testowałem poprzez rysowanie płyt gwoździem, robili to inni i program jest chwalony, natomiast chciałem zwrócić uwagę: aby go użyć, nie można mieć wiedzy typu *kliknij i zapomnij*.

Współcześnie wyrobiono w ludziach przekonanie, że komputer jest urządzeniem do dostarczania przyjemności. Tak, jest takie często powtarzane hasło "praca z tym, czy naszym programem, to przyjemność". Niestety, pomysł, by szybko i skutecznie za pomocą komputera wykonywać zadania, jakoś poszedł w odstawkę. Zasada przyjemności ma konsekwencje w tym, że obsługa programu ma się sprowadzać do kilku klików, najlepiej jednego. Zasadą jest, że user nie ma prawa niczego wiedzieć o komputerze. A jeśli, to ma być to wiedza kuchenna, przepis na wykonanie konkretnej jednej czynności, bo nie daj Panie Boziu, gdyby dwie myśli, to jak Kmicic nasz user byłby bliski utraty zmysłów. Cóż, chciałem na kilku przykładach pokazać, że warto wiedzieć. Po co? Bo wtedy sobie poradzimy.

Niestety stracimy i owszem, stracimy magiczne podejście do świata. Na przykład zaczniemy patrzeć na maszynę z ugryzionym jabłuszkiem nie jak na wyjątkową magiczną skrzynkę, ale będzie to kolejny komp, połączenie szeregu klamotów. Owszem, będziemy sobie radzić, ale zapewne nie będzie się nam chciało w podskokach, w środku nocy pędzić do kolejki w dniu premiery nowego tabletu. Niedobrze. W sumie tak niedobrze, że po przeczytaniu tego tekstu radzę spalić monitor. Proszę się nie dziwić, tak naprawdę świat tonie w oparach absurdu, więc porada jest konsekwentna.