

YILDIZ TEKNİK ÜNİVERSİTESİ FEN BİLİMLERİ ENSTİTÜSÜ

Bilgisayarlarda Bilgi Güvenliği

YÜKSEK LİSANS TEZİ

Banu Öntürk

1990

YILDIZ TEKNİK ÜNİVERSİTESİ
KÜTÜPHANE DOKÜMANTASYON
DAİRE BAŞKANLIĞI

Kot	R 368 54
Alındığı Yer	Fen Bil. Dns.
Tarih	13.4.95
Fatura	
Fiyatı	35.000,00
Ayniyat No	1-4
Kayıt No	51032
UDC	
Ek	

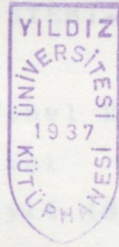
**YILDIZ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ**

YILDIZ ÜNİVERSİTESİ

D.B. No 48892

BİLGİSAYARLARDA BİLGİ

GÜVENLİĞİ



YÜKSEK LİSANS TEZİ

BİLGİSAYAR MÜH. BANU ÖNTÜRK

KÜTÜPHANE DOK. DAL. BAŞKANLIĞI

Y.T.O.

TEZ YÖNETİCİSİ

PROF. M. YAHYA KARSLIĞIL

İSTANBUL, 1990

1.1.1 Fonksiyon Çağrılarları	19
1.1.2 Dosya ve DOS Fonksiyonları	22
1.1.3 Veri Transferi ve Veri Transfer Alanları	32
1.1.4 Veri Transferi (DOS Memory Map)	32
Özet	39
Abstract	43
1. YAZILIMLAR NEDEN KOPYALANIYOR ?	45
1.1 Yazılımların kopyalanmasına Karşı Alınan Önlemler	1
1.2 Dünya Ülkelerinin Kopyalanmaya Karşı Aldıkları Ortak Tavır.	1
1.3 Yazılımların Gizli Kullanılmasının Sonuna Geliniyor.	3
2. DOS İŞLETİM SİSTEMİ	
2.1 DOS Hakkında Genel Bilgi	
2.1.1 DOS'un Yapısı	
2.1.1.1 İlk Kayıt (Boot Record)	4
2.1.1.2 Sadece Okunabilir Bellek BIOS Arabirimi	5
2.1.1.3 DOS Program File	5
2.1.1.4 Komut İşleyicisi (Command Processor)	5
2.1.2 DOS'un İlk Olarak Koşullandırılması	6
2.1.3 Uygun DOS Fonksiyonları	7
2.1.4 Veri Transfer Alanı (DTA)	7
2.2 DOS Disk Allocation	
2.2.1 DOS File Allocation Table (FAT)	9
2.2.1.1 12-Bitlik FAT Girişleri İçin	10
2.2.1.2 16-Bitlik FAT Girişleri İçin	10
2.2.2 DOS Disk Directory	
2.2.2.1 Directory Girişleri	11
2.3 DOS Keskileri ve Fonksiyonları	
2.3.1 DOS Saklayıcıları	13
2.3.2 Keskiler (Interrupts)	15

2.3.3 Fonksiyon Çağırımları	19
2.4 Kullanılan DOS Fonksiyon Çağırımları	22
2.5 DOS Kontrol Blokları ve Çalışma Alanları	
2.5.1 DOS Hafıza Haritası (DOS Memory Map)	37
2.5.2 DOS Program Segment	39
2.5.2.1 Program Segment Prefix	42
2.6 EXE File'lerinin Yapısı	
2.6.1 EXE File'lerinin Yapısı	44
2.6.2 Relocation Table	46
3. BİLGİSAYARLARDA BİLGİ GÜVENİRLİLİĞİ	
3.1 Bilgi Güvenliği Nedir ?	47
3.2 Bilgisayarlarda Bilgi Güvenliği Nasıl Sağlanır ?	48
4. YAZILIMLARIN GÜVENLİĞİ	
4.1 Yazılmış Olan Protect Programında Kullanılan Kullanılan Teknikler.	49
4.2 Program Hakkında Genel Bilgi	50
Programlar	53
Sonsöz	76
Kaynaklar	77
Özgeçmiş	78

ÖNSÖZ

Bilgisayar dünyasında ,donanım fiyatlarının giderek düşmesine karşılık ,yazılım fiyatlarındaki artış 'Yazılımın Korunması' konusunu gündeme getirmiştir. Diğer fikir ürünlerinde olduğu gibi, konu ne olursa olsun bir beyin gücü harcanarak ortaya çıkan bu ürünün de hiçbir emek harcamadan kopyalanarak kullanılması, bir donanımın para ödenmeden alınması ile daha açık bir deyimle çalınmasıyla eşdeğerdir. Yazılımların günden güne önem kazanması ve bu alanda çalışanların sayısının giderek artması , programcılarını kendi ürünlerini korumak için yoğun çalışmalara yönlendirmiştir.

Şimdiye kadar ,dünyanın farklı yerlerinde bu konu üzerinde çalışmaların yapılmasına karşılık net bir açıklama yapılabilmemiş ,ya da bu konuda yol gösterilebilmiş değildir.

Son zamanlarda ,tüm dünyada ve ülkemizde yazılım sektörünün en önemli sorunu olan kopyalamaya karşı ,ülkelerin düzenledikleri çeşitli toplantı ve faaliyetlerle ortak bir tavır alınmaya başlanılmıştır.

Yazılıma bilgisayarın kendisinden daha fazla değer verildiği şu günlerde yazılımın korunması önem kazanmaktadır. Yaptığım bu çalışmada,günümüzde çok yaygın kullanılan kişisel bilgisayarlar için yazılmış ve DOS işletim sistemi altında çalışan makinelerin disk ortamında kayıtlı bulunan yazılımların korunması amaçlanmaktadır.

Projemin gerçekleştirilmesinde bana katkıda bulunan değerli hocam Prof.M.Yahya Karslıgil'e teşekkür ederim.

YILDIZ ÜNİVERSİTESİ ,1990

BANU ÖNTÜRK

0251

ABSTRACT

ÖZET

Bu projenin sonunda oluşturulmuş olan ''Copy Protection'' adındaki program, kopyalanmasını istemediğimiz programların kaçak olarak kopyalanmasını önlemek amacıyla kullanılacaktır.

Herhangi bir programcı , yazmış olduğu programa koruma koymak istiyorsa kendi programını bu ''Copy Protection'' adını verdiğimiz programdan geçirmesi gerekecektir.

Copy Protection programındaki mantık,iki executive(Exe) programının olması.Bu executive programlardan biri ,koruma(protection) konulacak program , diğeri ise korumayı koyacak olan programdır. Sonuçta elimizde korunulmuş bir executive(Exe) program olmalıdır.

Artık elimizdeki bu executive(Exe) programı çalıştırabildiğimiz gibi bunun bir yabancı kullanıcı tarafından kopyalanıp başka bilgisayarda çalıştırılmasını engellemiş olmaktadır.

Bu programın yazılımında 8086 assembler dili kullanılmaktadır.Bu program, MS DOS ve PC DOS işletim sistemleri altında çalışabilecektir.

1. Yaxıllıqlar Neden Kopyalanıyır ?

1.1 Yaxıllıqların Kopyalanmasına Kərsi Alınan Gələcək

ABSTRACT

At the end of this project we obtained a program called 'Copy Protection'. This program will be used to prevent the programs that we don't want to be copied by other people.

If a programmer wants to put a 'Protection' to his program he has to use the Copy Protection with his own program so that at the end we will have a program which is protected. The logic in the Copy Protection Program is having two executive programs in it. One of these programs is the one which is called Copy Protection that protects the program and the other one is protected by it.

In conclusion we have to have an executive program which was protected.

No longer having the executive program worked we will also protect the program for being copied by someone else. 8086 assembly language is used in the writing of this program. This program works by the operating systems called MS DOS and PC DOS.

1. Yazılımlar Neden Kopyalanıyor ?

1.1 Yazılımın Kopyalanmasına Karşı Alınan Önlemler

Yazılım sanayi midir ? Bilginin, hizmetin maddi değeri nasıl ölçülmelidir. Hizmete para ödenmeli midir? gibi konularda kullanıcıların halen netleşmemiş soruları varken, ürün yada hizmet satıcı firmaların kimi zaman uyguladıkları satış ve hizmet biçimleri, bu bilinen kullanıcıların kamuoyunda yanlış yerleşmesine yol açmaktadır.

Uzun fikir emeği sürecinden geçerek üretilen yazılımların, bu emeğin hakettiği maddi değerini pratikte her zaman bulamaması ülkemizde halen yaşanan bir sorundur. Bilgisayar yazılımı satın alınması konusunda kullanıcı bilincinin gerektiği gibi yerleşmemiş olması bu sorunun süre gitmesinde önemli bir etken olmaktadır.

1.2 Dünya Ülkelerinin Kopyalanmaya karşı Aldıkları Ortak Tavrı

Başkalarının düşüncelerini kaynak göstermeden kullanmanın devlete karşı işlenmiş bir suç olduğu ilkesinin devletten önce vatandaşlar tarafından kabul edildiği Amerika Birleşik Devletler'inde başkasına ait bir yazılımı izinsizce kopyalayanlar tespit edildiği taktirde gazetelerde yayınlanıyor. Yazılımın korunması konusunun mikro yazılım pazarında odaklaştığı ülkede, ancak şu

sıralarda evrensel bir çözüme ihtiyaç duyuluyor. Fakat Amerika Birleşik Devletler'inde ana bilgisayar sistemlerine ait yazılım ürünlerinin korunması sadece telif hakkı(Copyright) ile sınırlı kalmayıp, ticari gizlilik ve normal kontrat yasalarıda kullanılmaktadır.

Fikir eserleri haklarının, kişiye değil kuruluşa ait olduğu görüşünü benimseyen İspanya'da telif hakkının (Copyright) korunma süresi 50 yıl ile sınırlanmıştır. Yakınlarda yazılımın korunması için yeni bir yasayı kabul eden İspanya parlamentosuna göre, telif hakkı altında kayıt isteğe bağlıdır ve aynı programın yeni versiyonları da kapsamaktadır.

İsviçre'de tüm yekili kopyalar ve kullanımda olanlar için 50 yıldır yürürlükte olan bir koruma yasası mevcuttur. Programların, bir sanat eseri gibi özel olarak tanımlanmasının yapılmadığı ülkede ticari gizlilik, rekabet ve patent yasaları yürürlüktedir. Bunlar elle tutulur gözle görülür ürünleri kapsadığı halde program tekniklerini ve gerçek kodlarını korumaktan uzak bir işleve sahiptirler.

Yukarıdaki açıklamalardan anlaşılacağı gibi bu ülkelerde yazılımın korunmasına karşı verilen önem uzun yıllar öncesine dayanıyor. En önemlisi, pek çok ülkede devlet konu ile ilgili geniş çaplı ve yeterli düzeyde önlem alma ve uygulama çabası içindedir.

Bütün bunlar programcılığın bu ülkelerde meslek olarak tanındığını göstermektedir. Ülkemizde programcılığın meslek olarak tanınmaması, üretilen yazılımların kopyalanmasına karşı etkili önlemlerin alınmamasına ve kanuna işlerlik kazandırma konusunda devletin yüzeysel bir tutum sergilenmesine neden olmaktadır.

Konuya yeterli önem verilmesi halinde yazılım sektörünün

daha da gelişerek Türkiye'ye yeni ufuklar kazandıracağıının en kısa zamanda anlaşılması gerektiğine inanıyorum.

1.3 Yazılımın Gizli Kullanılmasının Sonuna Geliniyor.

Standford'daki Software Security Şirketi yetkilileri tarafından üretilmiş olan bir alet, satın aldığımız programın yalnızca bizim, makinamızda çalışmasını sağlayarak ,bu korsan kopyalamaları önlemektedir.

Bu aletin çalışma prensibi, özel bir kod üretecek biçimde programlanmış bir yongaya dayanmaktadır. Korumalı yazılımların herbiri bir eşleme kodu içermektedir. Ve bu iki kod uyuşmadıkça, program çalışmamaktadır.

Korumayı (protect'i) sağlayan bu aletin yanı sıra programa koruma (protect) koyan PROLOCK adında bir programda mevcuttur. Bu programın yardımıyla, yazılmış olan bir programı korumaya almış olabiliriz.

2. DOS İşletim Sistemi (Kütüphane)

2.1.1 DOS'un Yapısı

DOS programı, bir IBMDOS.COM kütüğüdür. Bu program, kullanıcı programları için yüksek seviyeli bir aracıdır. DOS dört bölümden oluşur.

- İlk Kayıt (Boot Record) - Her diskteki bir değişikliğe
- Sadece Okunabilir Bellek BIOS Arabirimi
- DOS Program Kütüğü (IBMDOS.COM)
- Komut İşleyicisi (Command.COM)

2.1.1.3 Komut İşleyicisi (Command Processor)

2.1.1.1 İlk Kayıt(Boot Record)

COMMAND.COM komut işleyicisi, bu parçalarından oluşur.

İlk kayıt, DOS'un Format komutuyla formatlanmış bir diskette 0. yüz(side), 0. Track ve 0. Sektör den başlanmaktadır. Eğer, bilgisayarımızı üzerinde sistem yüklü olmayan bir diskette açmaya çalışırsak, karşımıza bir hata mesajı çıkar. Bunu sağlayan ilk kayıttır. Sabit Disklerde, ilk kayıt DOS bölgesinin ilk sektörüne yerleştirilir. Bu ilk kayıt için bir sektör kullanılır.

Bu kayıt, DOS'un başlatılması, bu kayıt (IBMDOS.COM) yapılır. Bu kayıt, bilgisayarın başlatılması ve abort, hata veya ilgili durumların değerlendirilmesini kapsar.

2.1.1.2 Sadece Okunabilir Bellek BIOS Arabirimi.

İlk Okunabilir Bellek Arabirimi, sadece bir kütüphane ve start-up programıdır. Bu kütüphane, AUTOEXEC.BAT kütüğüne bağlıdır. İlk başlatılmasından sonra, yukarıdaki programlar çalışır.

IBMBIO.COM kütüğü, sadece okunabilir bellek BIOS'a aracı bir modüldür. IBMBIO.COM, ROM BIOS cihaz yordamlarının, alçak seviyede bir aracıdır.

İlk başlatılmasından sonra, yukarıdaki komut işleyicisi çalışır.

2.1.1.3 DOS Program File(Kütük)

DOS programı, bir IBMDOS.COM kütüğüdür. Bu program, kullanıcı programları için yüksek seviyeli bir araçtır. IBMDOS.COM yönetim, disk yordamları için veri bloklama, tam bloklama ve built-in fonksiyonlarındaki bir değişikliğe kolayca erişebilmeyi amaçlamaktadır.

2.1.1.4 Komut İşleyicisi (Command Processor).

COMMAND.COM komut işleyicisi şu parçalardan oluşur.

1. Bir kalıcı parça IBMDOS.COM'u ve onun veri alanını takiben hafızaya yerleştirilir. Bu parça 22H (Bitirme Adresi), 23H (CTRL-Break Tutucusu) ve 24H (Kritik Hata Tutma) kesmelerini işlemek için yordamları, gerektiğinde geçici parçayı tekrar yüklemek bir yordamı kapsar.

Bütün standart DOS hata ayıklaması, bu parça içerisinde yapılır. Bu hata mesajlarını görüntülemesi ve Abort, Retry veya Ignore durumlarını cevaplandırmayı kapsar.

2. İlk koşullandırma kısmını, kalıcı bir kısım izler ve start-up esnasında kontrolü verir. Bu kısım AUTOEXEC.BAT kütüğünü içerir. İlk koşullandırma kısmı, yüklenen programın segment adreslerine karar verir. COMMAND.COM'un yüklenmesiyle bozulur.

3. Geçici bir kısımda hafızanın sonuna yüklenir. Dahili komut işleyicileri ve toplu kütük işleyicisini kapsayan başlı

başına bir komut işleyicisidir.

2.1.2 DOS'un İlk Olarak Koşullandırılması.

Sistem, açılarak veya Reset edilerek ilk olarak koşullandırılmış olur. BIOS ilk önce A disket sürücüsündeki ilk koşullandırma kaydına bakar. Eğer ilk koşullandırma kaydı bulunamazsa, ROM BIOS sabit diskteki, bölümü tarar. Eğer orada da bulamazsa 'Non System Disk ' hata mesajı gelir. Sistemin ilk olarak koşullandırılmasından sonra aşağıdaki işlemler sırasıyla yapılır.

- 1 - ilk koşullandırma kaydı hafızaya okunur ve kontrol verilir.
- 2 - ilk koşullandırma kaydı, ana directory'deki IBMBIO.COM ve IBMDOS.COM kütüklerinin olup olmadığına bakar. Bu iki kütükte ilk başta bulunmalı ve IBMBIO.COM başta bulunmalıdır.
- 3 - ilk koşullandırma kaydı IBMBIO.COM hafızaya yüklenir.
- 4 - IBMBIO.COM ilk koşullandırma kodu IBMDOS.COM'u yükler, çevre birimlerinin durumuna karar verir. Disk sistemini reset eder. Bağlı cihazları koşullandırır. Tesis edilen cihaz sürücülerini yükler. Alçak numaraları kesme vektörlerini belirler. IBMDOS.COM u aşağı doğru tekrar yerleştirir ve DOS'un ilk bayt'ını çağırır.
- 5 - 20H 'den 27H 'e kadar kesmeler için kesme vektörlerinin tablolarını, Program Segment Prefix 'i oluşturur.

6 - IBMBIO.COM, üst seviye komut işlemcisini yüklemek ve başlatmak için EXEC fonksiyon çağırmasını kullanır. Kabul edilen komut işleyicisi COMMAND.COM ' dur.

2.1.3 Uygun DOS Fonksiyonları

FAT, DOS tarafından bir file için disk üzerinde bir alan ayırır. Bu alan Cluster'dır. Bir file birden fazla DOS, kullanıcı programlarına, bütün kesme ve fonksiyonları sağlar. FAT DOS'un kütük yönetimi fonksiyonlarında kullanılabileceği iki tip fonksiyon çağırımı vardır.

- * Kütük kontrol bloğu (FCB) fonksiyon çağırması.
- * Gelişmiş fonksiyon çağırımları.

2.1.4 Veri Transfer Alanı(DTA)

DOS, bütün kütük okuma ve yazma işlemleri için hafızada belirli bir alan kullanır. Bu alan FCB (File Control Block) fonksiyon çağırımları ile kullanılır. DTA, buffer alanı olarakta adlandırılır. Bu alan, herhangi bir uygulama programının veri alanı içerisinde herhangi bir yerde olabilir ve program tarafından belirlenir. Bir anda sadece bir DTA kullanımda olabilir. COMMAND.COM tarafından bir program çalıştırıldığında, 'Program Segment Prefix'in içinde 80H de 128 bayt tutabilecek kadar büyük bir default DTA saptanır.

2.2 DOS Disk Allocation

2.2.1 DOS FILE Allocation Table(FAT)

FAT, DOS tarafından bir file için disk üzerinde bir alan ayırır. Bu alana Cluster denir. Bir file birden fazla cluster'a sahip olabilir. Bu cluster'lar birbirlerini takip etmek zorunda değildirlerdir FAT 12 veya 16 bitlik girişlere sahip olabilir. Disket'te Fat 'in genişliği 12 bittir. Hard Diskte ise 16 bittir.

FAT değerinin ilk byte 'ı şu değerlere sahiptir.

Hex Değeri	Anlamı
-----	-----
FF	Çift taraflı ve track başına 8 sector disket
FE	Tek taraflı ve track başına 8 sector disket
FD	Çift taraflı ve track başına 9 sector disket
FC	Tek taraflı ve track başına 9 sector disket
F9	Çift taraflı ve track başına 15 sector disket
F8	Hard disk.

12-bitlik Fat'lerde her bir giriş 3 hexadecimal karakterdir.
16-bitlik Fat'lerde ise her bir giriş 4 hexadecimal

karakterdir. Parantez içindeki her değer 16-bit lik Fat'lerde 4.sayıyı göstermektedir.

Cluster'in kullanılması
Bir kütüphane son cluster'ı olduğu
Gelecek Cluster'ın değerini gösterir.

000. Fat '1' bir kopya olarak kabul edilir.

2.2.1.3 12-bitlik FAT Dirisleri için



Directory sisteminde her bir dosya için bir cluster'ın başlangıcı belirtilir. Bu değer 0 ile 255 arasında olabilir. Eğer bu değer (FFF0-FFFF) ise bu kütüphane son cluster'ı olduğu anlamına gelir. Eğer bu değer (FFF0-FFFF) ise gelecekteki cluster'ın değerini gösterir.

2.2.1.4 12-bitlik FAT Dirisleri için

Directory sisteminde her bir dosya için bir cluster'ın başlangıcı belirtilir. Bu değer 0 ile 255 arasında olabilir.

Cluster sayısı 2 ile olabilir. Bu değer kütüphane son cluster'ı olduğu anlamına gelir. Eğer bu değer (FFF0-FFFF) ise bu kütüphane son cluster'ı olduğu anlamına gelir. Eğer bu değer (FFF0-FFFF) ise gelecekteki cluster'ın değerini gösterir.

Hex Değeri	Anlamı
-----	-----
(0)000	Cluster'ın kullanılmadığını
(F)FF8-(F)FFF	Bir kütüğün son cluster'ı olduğunu
(X)XXX	Gelecek Cluster'ın değerini gösterir.

DOS, FAT 'i iki kopya olacak şekilde oluşturur.

2.2.1.1 12-bitlik FAT Girişleri için

Directory girişinden kütüğün başlangıç cluster'ı okunur. Daha sonra Cluster sayısı 1.5 ile çarpılır. Bu değer saklayıcıya aktarılır. Eğer bu değer (FFF8-FFFF) ise bu kütüğün son cluster olduğunu gösterir. Eğer burdaki değer (FFF8-FFFF) den farklı ise gelecek cluster'ın değerini göstermektedir.

2.2.1.2 16-bitlik FAT Girişleri için

Directory girişinden kütüğün başlangıç cluster'ı okunur. Daha sonra ilgili işlemler yapılır.

Cluster sayısı 2 ile çarpılır. Bu değer saklayıcıya aktarılır. Eğer bu değer (FFF8-FFFF) ise bu kütüğün son cluster olduğunu gösterir. Eğer burdaki değer (FFF8-FFFF) den farklı ise gelecek cluster'ın değerini göstermektedir.

2.2.2 DOS Disk Directory

FORMAT komutu, ilk olarak bütün disklerde ana directoy'i oluşturur.

2.2.2.1 Directory Girişleri

Ana directory'den sonra gelen diğer directory'ler gerçek kütüklerdir. Bunların giriş sayısında hiçbir sınır yoktur. Bütün directory girişleri 32 byte uzunluğundadır. Directory'i girişleri aşağıdaki gibidir.

0-7. Baytlar

0'dan 7'ye kadar olan baytlar kütük ismini gösterir. Kütük isminin ilk baytı, kütük isminin durumunu içerir. Bir kütüğün durumu aşağıdaki durumları içerir.

- 00H Kütük ismi asla kullanılmamıştır.
- 05H Kütük isminin ilk karakteri gerçekten E5 dir.
- E5H Kütük ismi kullanılmıştır, fakat kütük silinmemiştir.
- 2EH Giriş bir directory içindir. Eğer ikinci bayt da 2EH ise, cluster alanı directory'inin parent directory sinin cluster numarasını kapsar.

8-10. Bayt'lar

Bu bayt lar kütüğün ismini kapsar.

11. Bayt

Bu bayt kütüğün niteliğini belirtir.

- 01H Kütüğün sadece okunabildiğini gösterir.
- 02H Kütüğün gizli olarak tutulduğunu belirtir.

- 04H Kütüğün bir sistem kütüğü olduğunu belirtir.
- 08H Girişin ilk 11 baytında volume etiketi bulunduğunu belirtir. Giriş için başka kullanılabilir bilgi içermez. Sadece kök directory'ide bulunur.
- 10H Girişin bir alt directory'i de tanımlandığını belirtir ve normal directory aramalarından hariç tutulur.
- 20H Bir arşiv bitini gösterir. Bit, kütüğe yazılıp kapandığında '1' yapılır. Bu, BACKUP ve RESTORE komutları tarafından kütüğün son kopyası alındığından beri değiştirilip değiştirilmediğine karar vermek için kullanılır.

Genel Saklayıcılar

12-21. Baytlar

Bu alan DOS tarafından kullanılır.

22-23. Baytlar

Bu baytlar, kütüğün yaratıldığı veya son olarak düzeltildiği zamanı kapsar.

24-25. Baytlar

Bu baytlar, kütüğün yaratıldığı veya son olarak düzeltildiği tarihi kapsar.

26-27. Baytlar

Bir kütüğün ilk cluster'ının başlangıç değerini verir.

28-31. Baytlar

Bu alan kütüğün bayt olarak büyüklüğünü içerir.

2.3 DOS Kesmeleri (Interrupts) ve Fonksiyonları (Functions)

DOS, kesme ve fonksiyon çağrılarında aşağıdaki saklayıcıları kullanır.

2.3.1 Dos Saklayıcıları

Genel Saklayıcılar	
Saklayıcılar	Tanımlamalar
AX	Akümlatör
AH	Akümlatör ün yüksek anlamı 8 biti
AL	Akümlatör ün alçak anlamı 8 biti
BX	Taban (16 - bit)
BH	Taban nın yüksek anlamı 8 biti
BL	Taban nın alçak anlamı 8 biti
CX	Sayıc1 (16 - bit)
CH	Sayıc1 nın yüksek anlamı 8 biti
CL	Sayıc1 nın alçak anlamı 8 biti
DX	Veri (16 - bit)
DH	Veri nin yüksek anlamı 8 biti
DL	Veri nin alçak anlamı 8 biti
Bayraklar	OF,DF,IF,TF,SF,ZF,AF,PF,CF

Göstergeler (Pointers)		
Saklayıcılar	Tanımı	
SP	Yığın Göstergesi (16 - bit)	
BP	Taban Göstergesi (16 - bit)	
IP	Komut Göstergesi	
Extra Segment (16 - bit)		
Segment Saklayıcıları		
Saklayıcılar	Tanımı	
CS	Code Segment (16 - bit)	
DS	Data Segment (16 - bit)	
SS	Stack Segment (16 - bit)	
ES	Extra Segment (16 - bit)	

20H Program Durdurma

Bir programdan çıkış için 20H komutu kullanılır. Bu komut kullanıldığında, bütün kütük buffer'ları yığılabilir ve bütün işlemleri kapatılır. Bu komutu kullandığınız anda bütün işlemler biter. Kısaca 20H'i kullandığınız anda program, CS saklayıcısının programın PSF (Program Segment Prefix) değerinde olması beklenir.

Index	Saklayıcıları

Saklayıcılar	Tanımı

DI	Varış Index i (16 - bit)

SI	Yığın Index i (16 - bit)

SS	Stack Segment (16 - bit)

ES	Extra Segment (16 - bit)

2.3.2 Kesmeler(Interrupts)

Eğer herhangi bir kesme vektörünü incelemek veya belirlemek istersek bu amaçlar için kullanılan DOS fonksiyon çağırımlarını (35H ve 25H) kullanılabilir. DOS, 20H'den 3FH' e kadar olan kesme vektörlerini kendisi kullanır. Bu mutlak hafıza yerleri 80H - FFH 'in DOS tarafından kullanılması demektir. Kesme vektörleri aşağıda gösterildiği gibidir.

20H Program Durdurma

Bir programdan çıkmak için 20H kesmesi kullanılır. Bu kesme kullanıldığı vakit, bütün kütük buffer'ları yok edilir ve kütük tutucuları kapatılır. Bu kesmeyi kullanmadan önce bütün kütükler kapanır. Kesme 20H'i kullanmadan önce program, CS saklayıcısının programın PSP (Program Segment Prefix) değerinde olması sağlanır.

21H Fonksiyon isteği gibidir.

Bu fonksiyonlar, 2.4 bölümde anlatılacaktır.

24H Kritik Hata Tutucu Vektörü

DOS içinde kritik bir hata olduğunda, bir kesme 24H ile transfer edilir. Hata tutucuya girişte, eğer hata bir disk hatasıysa AH'nin 7. biti 0, aksi halde 1 olacaktır.

Hata kodları aşağıdaki gibidir.

Hata kodları	Hatanın adı
0	Disket write-protect lidir
1	Bilinmeyen bir birim
2	Sürücü hazır durumda değil
3	Bilinmeyen bir komut
4	Veri hatası (CRC)
5	İstenen yapının genişliği yanlış
6	Arama hatası
7	Bilinmeyen ortam tipi
8	Sector okunamadı
9	Printer hatası
A	Yazma hatası
B	Okuma hatası
C	Genel hata

25H Mutlak Disk Okuma

Kontrolü direk olarak cihaz sürücüsüne transfer eder. Dönüşte orjinal bayraklar halen yığındadır. Bizden istenenler aşağıdaki gibidir.

AL = Sürücü sayısı (0 = A , 1 = B ve 2 = C)
 CX = Okunacak sector sayısı
 DX = Başlangıç lojik sector 'ün değeri
 DS:BX = Transfer adresi

Segment saklayıcıları hariç bütün saklayıcılar bozulur. Transfer başarılı ise elde bayrağı CF=0 dır. Diğer halde 1 dir.

26H Mutlak Disk Yazma

25H kesmesi gibi çalışır. Sadece okuma yerine, yazma işlemini yapmaktadır.

27H Kalıcı Durumda Ol

Programlar tarafından kullanılan bu vektör, COMMAND.COM kontrolü tekrar kazandığında programın kalıcı olmasını sağlar. 31H fonksiyon çağırması, bir programı kalıcı durumda tutmak için tercih edilen bir metottur. Çünkü bu dönüş bilgisinin geçmesini sağlar ve 64K dan büyük programların kalıcı olmasına mücade eder. Kendi kendini ilk koşullandırdıktan sonra, program DS veya ES'in değerlerine göreli olarak son adresin bir fazlasını DX'de belirleyip, sonra kesme 27H'i çalıştırılmalıdır. DOS,0 zaman programı DOS'un bir ilavesi olarak kabul eder. Bu nedenle program diğer programlar çalışırken üstüne yazılmaz. Bu düşünce,

kalması gereken kullanıcı tarafından yazılmış kesme tutucuları gibi programları yüklemek için çok yararlıdır.

28H - 2EH DOS için Saklıdır.

Bu kesmeler DOS'un kullanımı için ayrılmıştır.

2.3.3 Fonksiyon Çağırımları

DOS, cihaz G/Ç, kütük yönetimi, tarih ve zaman fonksiyonları, diğer programların çalıştırılması ve diğerleri için birçok sayıda fonksiyon içerir.

Bunlar aşağıda gösterilmiştir.

Hexs	Değerler	Anlam
00		Program durdurma
		Program Terminate
1 - C		Geleneksel karakter cihaz G / Ç
		Traditional character device I / O
D - 24		Geleneksel kütük yönetimi
		Traditional File Management
25 - 26		Geleneksel cihaz dışı fonksiyonlar
		Traditional nondevice functions
27 - 29		Geleneksel kütük yönetimi
		Traditional file management
2A - 2E		Geleneksel cihaz dışı fonksiyonlar
		Traditional nondevice functions
2F - 38		Genişletilmiş fonksiyon grubu
		Extended functions group
39 - 3B		Directory grub
		Directory Group
3C - 46		Genişletilmiş kütük yönetimi grubu
		Extended file management group
47		Directory grubu
		Directory Group

48 - 4B	Genişletilmiş hafıza yönetim grubu	
	Extended Memory Management Groub	

4C - 4F	Genişletilmiş fonksiyon grubu	
	Extended Function Groub	

54 - 57	Genişletilmiş fonksiyon grubu	
	Extended Function Groub	

59 - 5C	Genişletilmiş fonksiyon grubu	
	Extended Function Groub	

5E - 5F	Ağ fonksiyon grubu	
	Network Function Groub	

62	Genişletilmiş fonksiyon grubu	
	Extended Function Groub	

2.4 Kullanılan DOS Fonksiyon (Function) Çağrılmaları.

DOS fonksiyonlarından, proje gereği kullanılanlar aşağıdaki fonksiyonlardır.

OEH Diski Seç (Select Disk)

Amaç: DL 'de belirtilen disk ünitesine göre seçimi yapar.

Girişler	Saklayıcı İçerikleri

AH	OEH

DL	Sürücü numarası(0 = A,1 = B

Dönüşte	Saklayıcı İçerikleri

AL	Toplam sürücü sayısı

16H Kütük Yarat (Create File)

Amaç: Yeni bir file yaratır. ~~Block olarak çalışmaz~~

Girişler	Saklayıcı içerikleri	

AH	16H	

DS:DX	FCB 'un pointer'ı	

Dönüşte	Saklayıcı içerikleri	

AL	00H Kütük yaratıldı	
	FFH Kütük yaratılamadı	

1AH Disk Transfer Adresini Belirler

Amaç: Disk transfer adresini DS:DX olarak belirler.

Girişler		
Saklayıcı içerikleri		
AH	1AH	
DS:DX	Disk transfer adresi	
Dönüşte		
Saklayıcı içerikleri		
	None	

O FH Kütük Açma (Open File)

Amaç: Kütük açar.

Girişler		Saklayıcı İçerikleri

AH		O FH

DS:DX		FCB 'unun adresi

Dönüşte		Saklayıcı İçerikleri

AL		00H kütük açılmışsa
		FFH kütük açılmamışsa

10H Kütük Kapama (Close File)

Amaç: Açılmış bir kütüğü kapatır.

Girişler	Saklayıcı içerikleri	

AH	10H	

DS:DX	FCB 'unun adresi	

Dönüşte	Saklayıcı içerikleri	

AL	00H kütük bulundu	
	FFH kütük bulunamadı	

26H Yeni program segmenti yaratır(Create New Program Segment)

Amaç: Yeni program segmenti yaratır.

Girişler	Saklayıcı İçerikleri	

AH	26H	

DX	Yeni program segmenti için	
	segment sayısı	

Dönüşte	Saklayıcı İçerikleri	

	None	

2FH Disk Transfer Adresini Getir (Get Disk Transfer Address(DTA))

Amaç: Kullanılır durumdaki disk transfer adresini getirir.

Girişler	Saklayıcı İçerikleri	

AH	2FH	

Dönüşte	Saklayıcı İçerikleri	

ES:BX	Mevcut DTA	

Dönüşte	Saklayıcı İçerikleri	

AH	CF = 1 ise hata kodu	

	CF = 0 ise disk tutucu kodu	

3CH Bir Kütük Yaratır (Create a File)

Amaç: İsmi vererek yeni bir kütük yaratır.

Girişler	Saklayıcı İçerikleri	

AH	2CH	

DS:DX	Kütük isminin adresi	

CX	Kütük özelliği	

Dönüşte	Saklayıcı İçerikleri	

AX	CF = 1 ise hata kodu	
	CF = 0 ise kütük tutucu kodu	

3DH Bir Kütüğü Açar (Open a File)

Amaç: Belirlenmiş bir kütüğü açar.

Girişler	Saklayıcı İçerikleri	

AH	2DH	

DS:DX	Kütük isminin adresi	

AL	Erişim kodu	
	AL = 0 Okuma için açılıyor.	
	AL = 1 Yazma için açılıyor.	
	AL = 2 Okuma/Yazma için	
	açılıyor.	

Dönüşte	Saklayıcı İçerikleri	

AX	CF = 1 ise hata kodu	
	CF = 0 ise kütük tutucu kodu	

3EH Bir Kütüğü Kapatır. (Close a File Handle)

Amaç: Belirlenmiş bir kütüğü kapatır.

 | Girişler | Saklayıcı İçerikleri |

| AH | 2EH |

| BX | Açılmış yada yaratılmış bir |
 | | kütüğün file tutucusu |

 | Dönüşte | Saklayıcı İçerikleri |

| AX | CF = 1 ise hata kodu |
 | | CF = 0 ise kütük tutucu kodu |

3FH Bir Kütükden veya Cihazdan Okuma (Write to a File or Device)
(Read from a File or Device)

Amaç: Bir kütüğün istediğimiz kadarını buffer'a yükler.

Girişler		Saklayıcı İçerikleri

AH		3FH

BX		Kütük tutucu

DS:DX		Buffer adresi

CX		Okunacak bayt sayısı

Dönüşte		Saklayıcı İçerikleri

AX		CF = 1 ise hata kodu
		CF = 0 okunan bayt sayısı

40H Bir Kütükden veya Cihazdan Yazma (Write to a File or Device)

Amaç: Bir kütüğün istediğimiz kadarını buffer'a yükler.

Girişler	Saklayıcı İçerikleri	

AH	40H	

BX	Kütük tutucu	

DS:DX	Buffer adresi	

CX	Yazılacak bayt sayısı	

Dönüşte	Saklayıcı İçerikleri	

AX	CF = 1 ise hata kodu	
	CF = 0 yazılan bayt sayısı	

48H Hafızayı Ayırma (Allocate Memory) *Executa a Program*

Amaç: İstenildiği kadar hafızada yer ayrılır. *Memory*

Girişler	Saklayıcı içerikleri	

AH	48H	

BX	Paragraf sayısı	

Dönüşte	Saklayıcı içerikleri	

AX:0	Hafıza bloğunun işaretlenmesi	

AX	CF = 1 Hata kodları	

BX	Ayrılmış boğun genişliği	

4BH Programı yükle ve çalıştır. (Load or Execute a Program)

Amaç: Hafızaya yeni bir program yükler ve çalıştırır.

Girişler	Saklayıcı içerikleri	

AH	4BH	

DS:DX	Yüklenecek programın ismi	

ES:BX	Yüklenecek prg. parametresi	

AL	Fonsiyon değeri	

Dönüşte	Saklayıcı içerikleri	

AX	CF = 1 hata kodu	
	CF = 0 None	

62H Program Segment Prefix'i Okur. (Get Program Segment Prefix Address)

Amaç: Programın prefix adresini dönüş olarak kullanmak.

| Girişler | Saklayıcı İçerikleri |

| AH | 62H |

| Dönüşte | Saklayıcı İçerikleri |

| BX | Segment adresi |

2.5 DOS Kontrol Blokları ve Çalışma Alanları

2.5.1 DOS Hafıza Haritası(DOS Memory Map)

Yer	Kullanım
0000:0000	Kesme vektör tablosu
0040:0000	ROM iletişim alanı
0050:0000	DOS iletişim alanı
XXXX:0000	IBMBIO.COM - ROM I/O yordamlarına DOS aracısı
XXXX:0000	IBMDOS.COM - DOS kesme tutucuları hizmet yordamları
XXXX:0000	DOS buffer ları,kontrol alanları ve bağlanmış cihaz sürücüleri
XXXX:0000	COMMAND.COM 'un kalıcı parçası - 22H , 23H,24H kesmeleri için ve geçici parçayı yüklemek için gerekli kod
XXXX:0000	Dış komutlar (.COM veya .EXE kütükler)
XXXX:0000	.COM kütükleri için dış komutlar
XXXX:0000	Command.COM'un geçici bölümü

Notlar :

1. Hafıza haritası adresleri segment : ofset formatındadır.
örneğin : 0070:0000 adresi ,0700H mutlak adresidir.

2. DOS işletim alanı aşağıdaki gibi kullanılır.
0050:0000 Ekranı basma durum bayrağı.

0 Ekranı basma aktif değil veya başarılı ekran
basma işlemi.

1 Ekran basma çalışır durumda.

255 Ekranı basma basma işlemi sırasında hata
oluşturdu.

0050:0001 Basic tarafından kullanılır.

0050:0004 Tek sürücü modu durum baytı.

0 A sürücüsü için disket son kullanıldı.

1 B sürücüsü için disket son kullanıldı.

0050:0010-0021 BASIC tarafından kullanılır.

0050:0022-002F Disketin ilk koşullandırılması için
DOS tarafından kullanılır.

0050:0030-0033 MODE komutu tarafından kullanılır.

0050:0000 dan başlayan 256 bayt içerisindeki bütün yerler
DOS'un kullanımı için saklıdır.

3. Kullanıcı hafızası, mümkün hafızanın en düşük yerinden
kullanılır.

2.5.2 DOS Program Segment

Bir harici komut girildiğinde veya EXEC fonksiyon çağırması yoluyla bir program çağrıldığında, DOS programın başlayacağı hafızanın başı olarak kullanılacak en düşük adrese karar verir.

Program Segment içerisindeki 0 ofsetinde, DOS program segment prefix kontrol bloğunu oluşturur. EXEC programı 100H ofsetine programı yükler ve kontrolü verir.

Bir program kontrolü aldığı anda aşağıdaki durumlar oluşur:
Bütün programlar için:

- * Geçen enviroment'lerin segment adresi, Program Segment Prefix'in içinde 2CH ofsetinde yer alır.
- * Program Segment Prefix'in 50H ofsetinde DOS fonksiyonlarını çalıştırmak için gerekli kod vardır. Böylece, AH 'a istenen fonksiyon numarasını yerleştirerek, INT 21H komutunu işletmektense bir DOS fonksiyonunu işletmek için PSP + 50H bir uzun çağırma yapabilir.
- * Disk Transfer Adresi (DTA) 80H olarak belirlenir.
- * Komut çalıştırıldığında girilen ilk iki parametreye göre 5CH ve 6CH deki kütük kontrol blokları formatlanır. Eğer kütük ismi bir izi kapsıyorsa FCB sadece uygun bir sürücüyü kapsayacaktır.
- * 81H'deki formatlanmamış bir parametre alanı komut isminden sonra girilen karakterleri, 80H'deki alan karakter sayısını kapsar.
- * .COM kütükleri için, ofset 6, segment içinde kullanılabilir bayt sayısını kapsar.

* AX saklayıcısı ilk iki parametrede verilen sürücü isimlerinin uygunluğunu aşağıdaki gibi gösterir:

- ilk parametre uygunsuz bir sürücü göstergesini kapsıyorsa
AL = FFH aksi halde AL = 00H değerini alır.

- ikinci parametre uygunsuz bir sürücü göstergesini kapsıyorsa AH = FFH aksi halde AH = 00H değerini alır.

.EXE programları için:

* DS ve ES saklayıcıları Program Segment Prefix'i gösterecek şekilde belirlenir.

* CS,IP,SS ve SP saklayıcıları LINKER tarafından geçilen değerlerle belirlenirler.

.COM programları için:

* Bütün dört segment saklayıcısı, Program Segment Prefix kontrol bloğuyla başlayan ilk yerleştirme bloğunun segment adresini kapsarlar.

* Bütün kullanıcı hafızası programa ayrılır. Eğer program başka bir programı çalıştırmak isterse ,programı çalıştır- (4AH) fonksiyon çağırması yoluyla bazı hafıza bölgelerini serbest bırakmalıdır.

* Komut göstergesi (IP) 100H olarak belirlenir.

* SP saklayıcısı programın segmentinin sonunu gösterir. Ofset 6'daki segment büyüklüğü paragraf büyüklüğüne indirilerek yuvarlanır.

* Yığının başına bir word 0 yerleştirilir.

2.5.2.1 Program Segment Prefix.

00	01	02	03	04	05	06	07
	INT 20H	Hafızanın Başı	Saklı	DOS'a uzun çağırma			
				segmentte x bayt			
08	09	0A	0B	0C	0D	0E	0F
Uzun çağ. seg.	Durdurma adresi	Durdurma adresi	Ctrl-Break				
devamı	(IP)	(CS)	Adr. (IP)				
10	11	12	13	14	15	16	17
Ctrl-Break Adr.	Kritik Hata Çıkış Adresi						
(CS)	(IP)	(Cs)					
18	19	2A	2B	2C	2D	2E	2F
	Saklıdır			Env'nin seg.	Saklıdır		
				Adresi			
30				40			4F
				Saklıdır			
50	51	52	53	54	55	56	57
DOS çağırması				Saklıdır			
58	59	5A	5B	5C	5D	5E	5F
	Saklıdır			Açılmamış Standart FCB1			
60	61	62	63	64	65	66	67

2.6 EXE File'ların Yapısı ve Yüklmesi

2.6.1 EXE File'ların Yapısı

Linker programı tarafından oluşturulan EXE programları iki bölümden oluşur,

- Kontrol ve Relocation bilgisi
- Load Modülü

Exe programının başında Header adı verilen bir bölüm vardır.

Header aşağıdaki gibi bir formata sahiptir.

Hex değeri	İçerikleri
00 - 01	4DH,5AH Exe file olduğunu işaret eder.
02 - 03	Load modülünün 512 ye göre modundan kalan.
04 - 05	Header dahil, toplam sayfa sayısı
06 - 07	Relocation item sayısı.
08 - 09	Header'ın paragraf cinsinden değeri.
0A - 0B	Minimum paragraf sayısı.

0C - 0D	Maksimum paragraf sayısı.	

0E - 0F	Stack Segment'in paragraf	
	cinsinden offseti.	

10 - 11	Modül kontrolü aldığı zaman	
	SP offset değeri.	

12 - 13	Checksum değeri.	

14 - 15	Modül kontrolü aldığı zaman	
	IP offset değeri.	

16 - 17	Code Segment'in offset değeri.	

18 - 19	İlk relocation tablosunun	
	başlangıç offseti.	

1A - 1B	Taşma sayısı.	

2.6.2 Relocation Table

Header'ın 18H adresinde Relocation Table'ın başlangıç offseti belirtilir. Belirtilen bu offset değerinden itibaren de Relocation Item lar başlar. Yukarıdaki tabloda görüldüğü gibi herbir word 'ün ayrı değeri ve işlevi vardır.

3. Bilgisayarlarda Bilgi Güvenirliliği

3.1 Bilgi Güvenliği Nedir ?

Bilgi güvenliği genel olarak, şirket bilgilerinin herhangi bir nedenle açığa çıkmasının, değişmesinin ya da kaybolmasının engellenmesi olarak tanımlanabildiği gibi, bilgisayar ortamında da bilgilerin güvenliği olarak görülmektedir. Bilgisayarların günümüzde oldukça yaygın olarak kullanılması ve bilgisayarlaşmanın hemen hemen her ortama girmesi ile bilgi güvenliği daha değişik boyutlar kazanmıştır.

Sonuç olarak, günümüzde bilgisayarların yaşantımıza yoğun bir şekilde girmesi ile bilgi güvenliğinin sınırları oldukça genişlemiş, ciddi önlemler alınma aşamasına çoktan gelinmiştir. Yurtdışında bilgisayarlaşmaya bizden çok daha önceleri geçildiğinden bilgi güvenliği ile ilgili önemli gelişmeler kaydedilmiştir. Ülkemizde de, bilgisayar kullanımının yaygınlaşmasına paralel olarak konunun önemi anlaşılmaya ve gerekli önlemler alınmaya başlanmıştır. Bu arada, şunu da vurgulamak gerekir ki, sözkonusu edilen her ne kadar bilgisayar ortamındaki bilgi güvenliği ise de, diğer ortamlardaki bilginin güvenliğinin sağlanması da gözardı edilmemesi gereken bir husustur.

3.2 Bilgisayarlarda Bilgi Güvenliği

Bilgisayarlarda bilgi güvenliğini beş ayrı gruba ayırıp inceleyebiliriz.

* Yazılım Güvenliği

- Erişim Kontrol
- Password Sistemi
- Sistem Yazılım Kaynaklarının Güvenliği

- Yedekleme Güvenliği
- Programların, Uygulamaların Güvenliği
- Bilgisayar suçları ve Virüsler

* Donanım Güvenliği

- Donanım Teknik Güvenlikleri

* İletişim Güvenliği

- Bilgi Hatları Güvenliği
- Network Güvenliği
- Dial-Up Güvenliği

* Fiziksel ve Çevre Güvenliği

- Erişim Kontrol
- Doğal Afetler
- Enerji, Sıcaklık, Havalandırma, Nem

* İdari ve Organizasyonel Güvenlik

- Personel Güvenlik Sistemi
- Şirket Bilgi Güvenlik Sistemi
- Erişim
- Saklama
- Yok etme
- Dağıtım

Etkin bir güvenlik sistemi kurabilmek için, üst yönetimin konunun önemini gayet iyi anlaması, her türlü finansal ve yönetsel desteği vermeye hazır olması gerekir. Üst yönetimin bu desteğe hazır duruma gelmesi, bilgi işlem departmanlarının sorumluluğundadır. Bilgi güvenliğinin sağlanması için, güvenlik stratejisi ve planların geliştirilmesi ve uygulanması şarttır.

4. Yazılımların Güvenliği

Yazılımlara verilen önem, dünyada yaygınlaşmaya başlamasına rağmen, ülkemizde konu olarak yeni yeni gündeme gelmeye başlamıştır. Bu konuda yazılımcılar arasında bir birlik olmalıdır. Ülkemizde bu konuda fazla çalışma yapılmamasının sebebi dışarıya program yazan kuruluşların az olmasından kaynaklanmaktadır. Herhangibir yazılımın ortaya çıkması, hem zihin gücünü, hemde zamanı almaktadır. Bizler için önemli olan bu iki şeyin karşılığını bulamaması acı verici bir durumdur. Tezimin konusu bilgisayarlarda bilgi güvenilirliği olup, bu konunun içinde büyük önem taşıyan yazılımların korunması konusuna değindim. Yazılımların güvenliği nasıl sağlanmalıdır ? Yazılımlara nasıl kilit vurulabilir ?

Bu programı gerçekleştirirken amacım gizli olarak kopyalanan bir programın, protect konulan ortamdaki, farklı bir ortamda çalıştırılmasını engellemektir. Çalışmalarım sonucunda bunu gerçekleştirdim. Fakat her korumayı, zayıf düşürecek bir başka programın çıkacağını unutmamamız gerekir.

4.1 Yazılan Protect Programında Kullanılan Teknikler

Yazılmış olan, protect programında kullanılan teknik, protect konulacak programın executive (exe) haliyle, protect koyan programın executive (exe) halini birleştirerek, tek bir executive file haline getirmektir. Biz, oluşturduğumuz son executive file '1 çalıştırdığımızda, ilk önce ilgili protect işleminin kontrolü yapılacak, bu kontrol olumlu ise program çalışacak, aksi taktirde program çalışmayacaktır. Konunun başında da belirttiğim gibi, kopyaladığımız bu program, eger

protect konulmuş bir program ise, sadece protect konulan bilgisayarda çalışabilecektir. Aynı durum disketler içinde geçerlidir.

4.2 Program Hakkında Genel Bilgi

Program hakkında genel bir bilgi vermeden önce, önemli bir ayırımdan bahsetmek gerekecektir. Protect konulacak programımız iki farklı ortamda olabilir. Bunlardan birincisi Hard disk, ikincisi ise Flopy diskidir. Her ikisinde de izlenen yol aynı olmakla birlikte, kullanılan protect programları farklıdır. Protect konulacak programımız, Hard diske yerleşecek ise, programda izlenecek yol şöyledir. İlk olarak, yazılmış programın yardımıyla Partition Table 'ın bittiği yerden, Boot Record'un başladığı sectöre kadar olan alan içerisinde herhangi bir sectöre bir değer yazıp saklamak gerekmektedir. Burada sakladığımız bu değer, ileride bizim kontrolü bir exe'den diğerine verirken kullanacağımız adres değerlerinin durumuna göre, eklenecek veya çıkarılacaktır. Daha sonraki aşamada Join adını verdiğimiz program geliştirilecektir. Join programının amacı, protect konulacak program ile, protecti koyacak programı ortak bir header altında birleştirmektir. Ortak header sözüyle bahsetmek istediğim şey, elimde öyle bir header olmalı ki, kontrolü istediğim zaman, istediğim programa verebilmeliyim. (Header, her executive file'in başında bulunan minumum 512 bytes (1 page) uzunluğunda bir bilgidir. Uzunluğu, kullandığı Relacation Item ların sayısına bağlı olarak değişmektedir. Bu uzunluk 512'in katlarını içermektedir. Header hakkında daha detaylı bilgi bölüm 2.6 anlatılmıştır.)

Join programının oluşturacağı header da, ilk olarak kontrol, protect koyan programa verilmelidir. Onun çalışması bittikten sonra, kontrol protect konulacak

programa geçmelidir. Bu programı çalıştırdığımızda ilk olarak protect programı çalışmalı ve ilgili kontrolü yapmalıdır. Kontrolün sonucu olumlu ise, protect konulmuş program çalışmalı, aksi halde çalışmamalıdır.

Yukarıda bahsettiğimiz bu yolu izlerken sırasıyla şu işlemler yapılmıştır. Header'a ilk olarak Protect programının CS ve IP değerleri yerleştirilir. Bu işlemi yapmadan önce protect konulacak programın CS ve IP değerleri memory'de saklanmalıdır. Bu CS değerinden, Partition Table da sakladığımız değeri çıkarırız. Elde ettiğimiz yeni CS değeri ile IP'nin değerini Protect programına göndermeliyiz. İlk olarak kontrolü alan Protect programı, Join'den gönderilen CS değerine, Partition Table dan okuduğu değeride ekleyerek, kontrolü CS:IP üzerinden protect konulacak programa verir. Eğer, Protect koyan program Partition Table dan yanlış yada farklı bir değer okuyup, okuduğu bu değeri CS e ekleyip, kontrolü bu CS:IP değerinden verirse, program çalışmaz. Buda sistemin kitlenmesine neden olur.

Aklımıza şöyle bir soru gelebilir. Join programında, CS değerinden çıkarılan adres karşı kullanıcı tarafından debug da izlenebilir mi? Bu sorunun cevabı hayır dır. Çünkü join programından kullanıcı habersizdir. Kullanıcının eline geçen Join programının sonucunda oluşan exe programdır. Header da CS ve IP değerlerinden başka bilgilerde değişmektedir.

Flopy üzerinde gerçekleştirilmiş olan protect işleminde ise, izlenen yol şöyledir. Protect li programı, saklayacağımız disketi hafif bir iğne darbesiyle çizip, bad sector oluşturalım. Daha sonra Norton Utility ve PcTools programlarının yardımıyla hangi sector'ün bad sector olduğunu öğrenelim. (Bir başka kişide kendi disketi üzerinde aynı işlemi tekrarlarsa, aynı sector e isabet ettirebilme olasılığı çok zayıftır.)

Flopy için yazılmış olan Protect programında, bad sector olarak oluşturulan alandan okuma yapılır, eğer bu alan

fiziksel olarak bozuxsa ilgili interrupt 'ın yürütölmesinden sonra CF set edilir. CF'in durumuna göre, ilgili adrese kontrol verilir. (CS:IP olarak) Diyelimki bozmuş olduğumuz bu disketten DiskCopy komutunun yardımıyla, bir başka diskete kopyasını aldık. Programı bu diskette çalıştırdığımızda, bad sector lerin aynen geçmiş olmasına rağmen, disket fiziksel olarak bozuk olmadığından CF tamamıyla ters olarak değer alır. Böylece bu programı, bu ortamda çalıştıramayız.

```
FloppyName : String = 'c:\tp5\new.exe';
```

```
Var
  I, Uzun : Integer;
  Prg, Parametre : String(30);
  Tip, ProgName : String(30);
  Tp, NewName, J : String(40);
  FullName : String(30);
  Seg : String(1);
  Dosya : Text;
  Regs : Registers;
  Bizi : Array[1..32] of Byte;
```

```
Procedure Pause;
```

```
Var
```

```
Ch : Char;
```

```
Begin
```

```
Ch:=ReadKey
```

```
End;
```

```
(* Programın başlangıcı *)
```

```
Begin
```

```
Assign(Dosya, 'Dosya.Dta');
```

```
Prg:='c:\tp5\new.exe';
```

```
Parametre := ' ';
```

```
Exec(Prg, Parametre);
```

```
(* Yeni 'San' yollanan değerler Dosya adındaki *)
```

```
(* Yile'den okunuyor. *)
```

```
Reset(Dosya);
```

```
Read(Dosya, Bizi);
```

```
(* III karakter 00 = HardDisk ve 01 = FloppyDisk *)
```

```
Tip:=Int(Trunc(Bizi[1..2]));
```

```
ProgName:=Copy(Tip,3,30);
```

```
Tip:=Tip;
```

```
If (Tip = 000) Or (Tip = '00') Then Begin
```

```
Prg:='c:\tp5\yazpar.exe';
```

```
Ch:=ReadKey;
```

```
Exec(Prg, Parametre);
```

```
End;
```


Program Menu1;

```
(*
(* Bu program Menu2.Exe programini cagirmaktadir.
(* Bu programdan menulerin yardimiyla protect
(* konulacak programin ismi ve hangi ortamda oldugu
(* hakkında bilgi alinacaktır.
(*
{$M $4000,0,0}
Uses Crt,Graph,Dos;
```

Label son;

Const

```
OldName : String = 'c:\tp5\new.exe';
FlopyName : String = 'c:\tp5\fnew.exe';
```

Var

```
I,Uzun : Integer;
Prg,Parametre : String[30];
Tip,ProgName : String[30];
Tp,NewName,J : String[30];
FullName : String[30];
Sec : String[1];
Dosya : Text;
Regs : Registers;
Dizi : Array[1..32] of Byte;
```

Procedure Pause;

Var

```
Ch : Char;
```

Begin

```
Ch:=ReadKey
```

End;

(* Programin baslangici *)

Begin

```
Assign(Dosya,'Dosya.Dta');
```

```
Prg:='c:\tp5\menu2.exe';
```

```
Parametre :='';
```

```
Exec(Prg,Parametre);
```

(* Menu2'den yollanan degerler Dosya adindaki*)

(* File'dan okunuyor. *)

Reset(Dosya);

Read(Dosya,Tip);

(* Ilk karakter HD = HardDisk ve FD = FloopyDisk *)

```
Tp:=Copy(Tip,1,2);
```

```
ProgName:=Copy(Tip,3,30);
```

```
Tip:=Tp;
```

```
If ((Tip = 'HD') OR (Tip = 'hd')) Then Begin
```

```
Prg:='c:\tp5\yazpar.exe';
```

```
Parametre:='';
```

```
Exec(Prg,Parametre)
```

```
End;
```



```

For I:=1 To 30 Do Begin
    Tp:=Copy(ProgName,I,1);
    If Tp = ' ' Then Goto son
End;

son:
ProgName:=Copy(ProgName,1,I);
Prg:=' ';
Prg:=Concat(Prg,ProgName);
Parametre:=Prg;

If ((Tip = 'HD') OR (Tip = 'hd')) Then
    Prg:='c:\tp5\join.exe'
    Else Prg:='c:\tp5\fjoin.exe';

Uzun:=Length(Parametre);
NewName:=Copy(Parametre,2,Uzun-1);
J:='.exe';
NewName:=Concat(NewName,J);
Uzun:= Length(NewName);

For I:= 1 To 30 Do Dizi[I] := 0;
For I:= 1 To Uzun Do Dizi[I] := Ord (NewName[I]);

Exec(Prg,Parametre);
Close(Dosya);

(* Son Ekran *)

Prg:='c:\tp5\menu3.exe';
Parametre:='';
Exec(Prg,Parametre);

Assign(Dosya,'Dosya.Dta');
Reset(Dosya);
Read(Dosya,FullName);
Close(Dosya);

(* Protect konulan Programin ismi su an New.Exe olarak*)
(* gecmektedir. New.Exe'ye eski.exe'nin ismini vermeden,*)
(* oncekini silmemiz gerekir *)

Sec:=Copy(FullName,1,1);

If ((Sec = 'E') OR (Sec = 'e')) Then Begin

    Regs.ah:= 65;           {Delete a file }
    Regs.ds:= Seg(Dizi); { DS:DX   Protect konulan File }
    Regs.dx:= Ofs(Dizi[1]);
    Intr($21,Regs);

    If ((Tip = 'HD') OR (Tip = 'hd')) Then Begin

        Regs.ah:= 86;           { Rename a file }
        Regs.ds:= Seg(OldName); { DS:DX   New.Exe }
        Regs.dx:= Ofs(OldName[1]);
        Regs.es:= Seg(Dizi);    { ES:DI }
        Regs.di:= Ofs(Dizi[1]);
    
```



```

Intr($21,Regs);

End

Program Menu2;
(*
(* Bu program ile kullanicidan bazi bilgiler
(* aliniyor
(* Bu file
(* ve hangisi
(* gosteriliyor
(*
Uses Crt;

Label cavi;

End;

Var
If ((Sec = 'Y') OR (Sec = 'y')) Then Begin
    GraphDriver:=GraphMode;
    I,Tut
    Tip,Program
    Ch
    Prog,Parametre
    Kontrol
    Deger
    Regs
    Uzun:=Length(FullName);
    FullName:=Copy(FullName,2,Uzun-1);
    FullName:=Concat(FullName,J);
    Uzun:=Length(FullName);

    For I:=1 To 30 Do Dizi[I]:=0;
    For I:=1 To Uzun Do Dizi[I]:= Ord(FullName[I]);

    Procedure P;
    Var
    Ch: Char;
    Begin
    Ch:= ' ';
    Regs.ah:= 86;
    Regs.ds:= Seg(OldName); { DS:DX New.Exe }
    Regs.dx:= Ofs(OldName[1]);
    Regs.es:= Seg(Dizi); { ES:DI }
    Regs.di:= Ofs(Dizi[1]);
    Intr($21,Regs);
    End
    Else Begin

    Regs.ah:= 86;
    Regs.ds:= Seg(FlopyName); { DS:DX New.Exe }
    Regs.dx:= Ofs(FlopyName[1]);
    Regs.es:= Seg(Dizi); { ES:DI }
    Regs.di:= Ofs(Dizi[1]);
    Intr($21,Regs);
    End
    End;

Assign(Dosya, 've.Dta');
Kontrol:=False;

End.

(* Arka Plan *)
SetColor(Blue);
SetFillStyle(SolidFill, Blue);
FloodFill(300,300,Blue);
Bar(0,0,400,400);

(* On Ren *)
SetColor(LightRed);
SetFillStyle(SolidFill, LightRed);
Bar(100,80,350,390);

SetColor(Red);
SetFillStyle(SolidFill, Red);
Bar(150,120,370,400);

(* Yuzeyin Ismi *)
SetTextStyle(1,0,0);SetColor(Cyan);
OutTextXY(150,420,'COPY PROTECTION ..... 1989');

```


Program Menu2;

```
(*
(* Bu program ile kullanicidan bazi bilgiler
(* aliniyor.
(* Bu bilgiler calismanin hangi ortamda yapildigini
(* ve hangi program uzerinde calisma yapacagimizi
(* gosterir.
(*)
```

Uses Crt, Graph, Dos;

Label cevir, a1, a2, c0;

Var

```
GraphDriver, GraphMode : Integer;
I, Tut : Integer;
Tip, ProgName : String[30];
Ch : Char;
Prg, Parametre : String[30];
Kontrol : Boolean;
Dosya : Text;
Regs : Registers;
```

Procedure Pause;

Var

Ch : Char;

Begin

Ch:= ReadKey

End;

(* Programin baslangici *)

Begin

```
(* NOTE : Graph kullanildigi programdan Exec komutu *)
(* ile herhangi bir islem yapamiyoruz.
(* Ayrica donuste Graph'in kapatilmasi gerekir. *)
```

GraphDriver:=Detect; InitGraph(GraphDriver, GraphMode, '');
DirectVideo:=False;

Assign(Dosya, 'Dosya.Dta');
Kontrol:=False;

(* Arka Fon *)

```
SetColor(Blue);
SetFillStyle(SolidFill, Blue);
FloodFill(300, 300, Blue);
Bar(0, 0, 640, 480);
```

(* On Fon *)

```
SetColor(LightRed);
SetFillStyle(SolidFill, LightRed);
Bar(100, 80, 550, 390);
```

```
SetColor(Red);
SetFillStyle(SolidFill, Red);
Bar(150, 120, 570, 400);
```

(* Projenin Ismi *)

```
SetTextStyle(1, 0, 0); SetColor(Cyan);
OutTextXY(60, 420, 'COPY PROTECTION ..... 1989');
```


cevir:

```
SetColor(Red);
SetFillStyle(SolidFill,Red);
Bar(150,120,570,400);
Bar(200,350,250,375);
SetTextStyle(1,0,1);SetColor(Cyan);

c0: OutTextXY(200,150,'Protect konulacak programiniz');
Regs.e OutTextXY(340,200,'hangi ortamda ...');
Regs.f SetColor(Yellow);
Regs.g OutTextXY(200,300,'Hard Disk : HD
Floopy Disk : FD');
Intr($1,Regs);

SetBkColor(Red);

(* Cursor set edildi. *)
Bar(150,370,500,390);
Regs.ah:= 2;
Regs.bh:= 0;
Regs.dh:= 23; (* Row *)
Regs.dl:= 20; (* Colon *)
Intr($10,Regs);

ReadLn(Tip);

(* Girilen deger siliniyor *)
Bar(150,370,500,390);

If ((Tip = 'HD') OR (Tip = 'hd')) Then Begin
    SetColor(LightRed);
    SetFillStyle(SolidFill,LightRed);
    Bar(310,300,345,325);
    SetColor(Yellow);
    OutTextXY(315,300,'HD');
    Kontrol:=True;
    Delay(1000);
    End;

If ((Tip = 'FD') OR (Tip = 'fd')) Then Begin
    SetColor(LightRed);
    SetFillStyle(SolidFill,LightRed);
    Bar(500,300,535,325);
    SetColor(Yellow);
    OutTextXY(505,300,'FD');
    Kontrol:=True;
    Delay(1000);
    End;

(* Eger yanlis veri girilmisse..... *)

If Kontrol = False Then Begin
    SetColor(Cyan);
    OutTextXY(200,350,'Yanlis veri ...');
    Delay(1000);
    Goto ceviri
End;

Kontrol:=False;
SetColor(Red);
```



```

        SetFillStyle(SolidFill,Red);
        Bar(150,120,570,400);
        SetTextStyle(1,0,1);SetColor(Cyan);

Program Menu;

OutTextXY(200,150,'Protect konulacak programinizin');
OutTextXY(340,200,'ismini giriniz ...');
OutTextXY(200,250,'Path vererek giriniz...');

c0:
Regs.ah:= 2;
Regs.bh:= 0;
Regs.dh:= 23; (* Row *)
Regs.dl:= 20; (* Coloun *)
Intr($10,Regs);

ReadLn(ProgName);
If ProgName = ' ' Then Goto c0;
Bar(150,370,500,390);
SetColor(LightRed);
SetFillStyle(SolidFill,LightRed);
Bar(280,300,500,330);
SetColor(Yellow);
OutTextXY(300,300,ProgName);

If ((Tip = 'HD') OR (Tip = 'FD')) Then
    OutTextXY(450,300,'.EXE')
Else OutTextXY(450,300,'.exe');

Delay(1000);
SetColor(Red);
SetFillStyle(SolidFill,Red);
Bar(150,120,570,400);
SetTextStyle(1,0,1);
SetColor(Cyan);

OutTextXY(430,300,'WAIT ...');

Delay(1000);

ReWrite(Dosya);
Write(Dosya,Tip);WriteLn(Dosya,ProgName);

Close(Dosya);
CloseGraph;
End.

```



```

Program Menu3;
(*
(* Bu program protect isinin bittigini gosterir *)
*)

Uses    Crt,Graph,Dos;

Label  cevir,yaz,yaz0;

Var
    GraphDriver,GraphMode    : Integer;
    Regs                      : Registers;
    EY                        : Char;
    Dosya                     : Text;
    FullName                  : String[30];

Procedure Pause;
Var
    Ch : char;
Begin
    Ch:=ReadKey
End;

(* Programin baslangici *)

Begin
    GraphDriver:=Detect; InitGraph(GraphDriver,GraphMode,'');

    DirectVideo:=False;
    Assign(Dosya,'Dosya.Dta');

    SetColor(Blue);
    SetFillStyle(SolidFill,Blue);
    FloodFill(300,300,LightGray);
    Bar(0,0,640,480);

    SetColor(LightRed);
    SetFillStyle(SolidFill,LightRed);
    Bar(100,80,550,390);

    SetColor(Red);
    SetFillStyle(SolidFill,Red);
    Bar(150,120,570,400);
    SetTextStyle(1,0,0);SetColor(Cyan);

    OutTextXY(60,420,'COPY  PROTECTION  .....  1989');

    SetColor(Red);
    SetFillStyle(SolidFill,Red);
    Bar(150,120,570,400);
    SetTextStyle(1,0,1);SetColor(Cyan);

    OutTextXY(200,250,'Protect  islemi
                        tamamlanmistir ...');

    Pause;
    SetColor(Red);
    SetFillStyle(SolidFill,Red);
    Bar(150,120,570,400);
    SetTextStyle(1,0,1);SetColor(Cyan);

```



```
OutTextXY(200,200,'Programinizi hangi
                                isimle kullanmak');
OutTextXY(200,250,'istiyorsunuz ? ...
                                Eski (E) / Yeni (Y)');
```

```
{Cursor set ediliyor }
```

```
cevir:
```

```
    SetBkColor(Red);
    Regs.ah:=2;
    Regs.bh:=0;
    Regs.dh:=23;
    Regs.dl:=20;
    Intr($10,Regs);

    ReadLn(EY);
    If ((EY = 'Y') OR (EY = 'y')) Then Goto Yaz;
    If ((EY = 'E') OR (EY = 'e')) Then Goto Yaz0;

    SetColor(Red);
    SetFillStyle(SolidFill,Red);
    Bar(150,350,450,400);
    SetColor(Yellow);

    OutTextXY(380,350,'YANLIS VERI');Delay(1000);
    SetColor(Red);
    Bar(380,350,500,400);
    Goto cevir;
```

```
Yaz:      SetColor(Red);
           Bar(380,350,500,400);
           SetColor(Red);
           SetFillStyle(SolidFill,Red);
           Bar(150,120,570,400);
           SetTextStyle(1,0,1);SetColor(Cyan);
```

```
OutTextXY(200,250,'Path ile birlikte yeni ismi
                                giriniz...');
    SetBkColor(Red);
```

```
    Regs.ah:=2;
    Regs.bh:=0;
    Regs.dh:=23;
    Regs.dl:=20;
    Intr($10,Regs);
```

```
    Read(FullName);
    SetColor(LightRed);
    SetFillStyle(SolidFill,LightRed);
    Bar(250,300,500,330);
    SetColor(White);
    OutTextXY(260,300,FullName);
```

```
Yaz0:
```

```
    SetColor(Red);
    SetFillStyle(SolidFill,Red);
    Bar(150,350,450,400);

    ReWrite(Dosya);
    Write(Dosya,EY);WriteLn(Dosya,FullName);
    Close(Dosya);
```



```

; Program Name : Protect
; -----
; Bu program herhangi bir file protect koymaktadır.
; -----
Codeseg      segment byte
Begin        proc far
Stackseg     Assume cs:Codeseg
;
;          jmp      devam
adres        dd      ?
buf          db      512 dup(?)
devam:
;          push     es
;          push     ds
;          mov      ax,cs
;          mov      ds,ax
;          mov      es,ax
;
;
; Partition Table'in Side = 0, Cylinder = 0 ve
; Sector = 2 okunuyor. Burda yazili olan deger
; Protect konulacak olan programin CS degerine
; eklenecek.
;
;          ..... INT 13H .....
;          Fixed Disk I/O Interface
;
;          mov      ah,2 ; Memory den sector okumak.
;          mov      al,1 ; Kac adet sector okunacak.
;          mov      ch,0 ; Okunacak Track'in degeri.
;          mov      cl,2 ; Okunacak sector'un degeri.
;          mov      dh,0 ; Head'in degeri.
;          mov      dl,80h ; Drive degeri.
;          mov      bx,offset buf
;          int      13h
;
;          pop      ds
;          pop      es
;
; Adres degiskeninden CS'in degeri okundu.
;
;          mov      ax,word ptr[adres+2]
;
; Partition Table daki deger ile toplandi.
;
;          add      ax,word ptr[buf]
;          mov      bx,ds ; PSP'nin baslangic degeri.
;          add      ax,bx ; CS degeri eklendi.
;          add      ax,10h ; Psp'nin uzunlugu eklendi.
;
; Adres degiskeninin Segment word'une eklendi.
;
;          mov      word ptr[adres+2],ax
;
; Protect konulacak Exe'nin basina dondu.
;
;          jmp      cs:adres
Begin        endp
Codeseg      ends
End          Begin

```



```

; Program Name : Yazpar
; Bu program Side = 0 , Cylinder = 0 ,
; Sector = 2 ye yazmaktadır.
; Partition Table dan sonraki sector.
;
Stackseg segment stack para 'stack'
                db 32 dup(?)
Stackseg ends

;
Dataseg segment para 'data'
buffer          db 512 dup(?)
Dataseg ends

;
Codeseg segment para 'code'
Begin          proc far
Assume cs:codeseg,ds:dataseg,ss:stackseg,es:dataseg
                push    ds
                sub     ax,ax
                push    ax
                mov     ax,dataseg
                mov     ds,ax
                mov     es,ax
;
                mov     ax,XXXX ; Saklanan deger.
                mov     word ptr buffer[0],ax
;
; Int 13H kullanilarak ilgili sector okunuyor.
; Assume cs:Codeseg,ss:Stackseg,ds:Dataseg
                mov     ah,3 ; Istenen sector'e yazilmasi
                mov     al,1 ; Okunacak sector sayisi
                mov     ch,0 ; Track'in degeri
                mov     cl,2 ; Okunacak sector'un degri
                mov     dh,0 ; Head'in degeri
                mov     dl,80h ; Drive degeri(80H for disk)
                lea     bx,buffer
                int     13h
;
                jnc     ok
;
ok:             ret
Begin          endp
Codeseg        ends
End begin

```



```

; Program Name : Join
; -----
; Bu program iki EXE file'i ortak Header
; altinda birlestirir.
; -----
;
Stackseg segment para stack 'stack'
        dw      32 dup(?)
Stackseg ends
Dataseg segment para
koru     db      30 dup(?)
exten    db      '.exe',0
buf1     db      512 dup(?)
hdl1     dw      ?
prtct    db      '\tp5\Protect.exe',0
buf2     db      512 dup(?)
hdl2     dw      ?
new db    'New.exe',0
buf3     db      512 dup(?)
hdl3     dw      ?
x1        dw      ?
x2        dw      ?
px1       dw      ?
px2       dw      ?
ofs       dw      ?
sgm       dw      ?
Dataseg   ends
Codeseg   segment para 'code'
Begin     proc     far
Assume cs:Codeseg,ss:Stackseg,ds:Dataseg
;
; Bu bolumde protect konulacak programin ismi
; disaridan giriliyor.
; Bu parametre PSP nin 80H location nina yerlesir.
; 80H de parametrenin uzunlugu ve bir bosluk
; degeri vardir.
; 81H de bosluk 82H den itibaren parametre yazilir.
;
        mov     si,82h
        mov     ax,Dataseg
        mov     es,ax
        lea     di,koru
        sub     cx,cx
        mov     cl,byte ptr [si-2]
        dec     cx
        cld
        ; Okumanin ileri dogru yapilmasi icin.
        rep     movsb      ; DS:SI ---> ES:DI
        mov     ds,ax      ; mov ds,dseg
;
; Parametrenin sonuna .Exe,'0' ekleniyor.
;
        lea     si,exten
        mov     cx,5      ; .Exe,'0' (5 chars.)
        rep     movsb
;
; Protect konulan programin ilk 512 bytes okunuyor.
;
        mov     ah,3dh ; File aciliyor.
        lea     dx,koru
        mov     al,0      ; Sadece okunabilir.
        int     21h

```



```

;
;      mov      hdl1,ax
;
;      call     read1
;
;      mov      ah,3dh ; File aciliyor.
;      lea      dx,prtct
;      mov      al,0
;      int      21h ; Sadece okunabilir.
;      mov      hdl2,ax
;
;      call     read2
;
;      mov      ax,word ptr buf2[4]
;      mov      px1,ax ; Header dahil...
;      mov      ax,word ptr buf2[2]
;      mov      px2,ax
;
;      mov      ax,word ptr buf1[4]
;      mov      x1,ax
;      mov      ax,word ptr buf1[2]
;      mov      x2,ax
;
; Korunulacak programin kalan bytes lari
; paragraf olacak bicimde
; duzunleniyor. Cunku Protect programina giderken
; IP = 0 olmalı.
;
;      mov      cx,16
;      div      cl
;      sub      cl,ah
;      add      x2,cx
;
;      mov      ax,x2
;      add      ax,word ptr buf2[2]
;      cmp      ax,512
;      jb       done
;      sub      ax,512
;      inc      word ptr buf1[4]
done:
;      mov      dx,word ptr buf2[4]
;      dec      dx
;      add      word ptr buf1[4],dx
;      mov      word ptr buf1[2],ax
;
; Protect konulacak file'in toplam uzunlugunun bulunmasi.
;
;      mov      bx,word ptr buf1[8]
;      mov      ax,x1
;      mov      cx,32
;      mul      cx
;      sub      ax,bx
;      mov      dx,x2
;      mov      cl,4
;      shr      dx,cl
;      add      dx,ax
;      sub      bx,bx
;
; Protect konulacak programin CS ve IP degerleri
; memory de saklaniyor.
; Bu degerler Protect programina gonderilecek.
; Adres degiskenine atanacak.
; Simdilik ofs ve sgm degiskenlerinde saklaniyor.
;
;      mov      ax,word ptr buf1[14h]
;      mov      cx,word ptr buf1[16h]

```



```

        mov     ofs,ax
        mov     sgm,cx
;
; Header'a Protect programinin CS:IP degerleri ataniyor.
; IP:0000 CS:Protect programini
; Load modulunun uzunlugu (parag.).
;
        mov     word ptr buf1[14h],bx
        mov     word ptr buf1[16h],dx
;
; NEW.EXE file'inin yaratilmasi.
; Protect.Exe ile protect konulacak herhengibir
; file'in exe'lerinin ortak Header altinda birlestirilmesi.
;
        mov     ah,3Ch
        lea     dx,new
        mov     cx,0
        int     21h
        mov     hdl3,ax
;
        mov     ah,3dh
        lea     dx,new
        mov     al,1
        int     21h
        mov     hdl3,ax
;
        mov     ah,40h
        mov     bx,hdl3
        lea     dx,buf1
        mov     cx,512
        int     21h
;
        mov     cx,x1
        dec     cx
tekrar:
        push    cx
        call    read1
        mov     ah,40h
        mov     bx,hdl3
        lea     dx,buf1
        mov     cx,512
        int     21h
        pop     cx
        loop    tekrar
;
        mov     ah,3fh
        mov     bx,hdl1
        lea     dx,buf1
        mov     cx,x2
        int     21h
;
        mov     ah,40h
        mov     bx,hdl3
        lea     dx,buf1
        mov     cx,x2
        int     21h
;
        call    read2
;
; Bu page'in 2,3,4 ve 5. byte larina adres degiskeni
; karsilik gelmektedir.
; Bu byte larin 2-3 word'u ----> Offset degerine
; 4-5 word'u ----> Segment degerine karsilik
; gelmektedir.

```



```

;
    mov     ax,ofs
    mov     word ptr buf2[3],ax
    mov     ax,sgm
    sub     ax,XXXXh
    mov     word ptr buf2[5],ax
;
; Adres degikenindeki bu deger ile Protect.Exe nin
; calismasindan sonra kontrol Protect konulan
; programa verilecektir.
;
    mov     cx,512
    mov     ah,40h
    mov     bx,hdl3
    lea     dx,buf2
    int     21h
;
    mov     cx,px1
    dec     cx
    dec     cx
tkr:
    push    cx
    call    read2
    mov     ah,40h
    mov     bx,hdl3
    lea     dx,buf2
    mov     cx,512
    int     21h
    pop     cx
    loop    tkr
;
    call    read2
    mov     cx,px2
    mov     ah,40h
    mov     bx,hdl3
    lea     dx,buf2
    int     21h
;
    mov     ah,3eh
    mov     bx,hdl1
    int     21h
;
    mov     ah,3eh
    mov     bx,hdl2
    int     21h
;
    mov     ah,3eh
    mov     bx,hdl3
    int     21h
;
    mov     ah,4ch
    int     21h
;
Begin    endp
;
read1    proc    near
    mov     ah,3fh
    mov     bx,hdl1
    lea     dx,buf1
    mov     cx,512
    int     21h
    ret
read1    endp
;

```



```

; Program name : Flopyaz
; -----
; Bu program herhangi bir file protect koymaktadır.
; -----
Codeseg      segment byte
Begin        proc      far
Assume       cs:Codeseg
;
;          jmp      devam
;          adres    dd      ?
;          buf      db      512 dup(?)
devam:
;          push     es
;          push     ds
;          mov      ax,cs
;          mov      ds,ax
;          mov      es,ax
;
;          .....   INT 13H   .....
;          Floppy Disk I/O Interface
;
;          sub      si,si
;          mov      ah,3 ; Memory den floppy'ye yazmak.
;          mov      al,1 ; Kac adet sector yazilacak.
;          mov      ch,15 ; Okunacak Track'in degeri.
;          mov      cl,4 ; Okunacak sector'un degeri.
;          mov      dh,0 ; Head'in degeri.
;          mov      dl,0h ; Drive degeri.floppy icin (0)
;          mov      bx,offset buf ; 4. sector
;          int      13h
;
;          Carry olusmussa disket fiziksel olarak bozuk
;
;          c0:      adc      si,4
;                  pop      ds
;                  pop      es
;
;          Adres degiskeninden CS'in degeri okundu.
;
;          mov      ax,cs:[si]
;
;          mov      bx,ds ; PSP'nin baslangic degeri.
;          add      ax,bx ; CS degeri eklendi.
;          add      ax,10h ; Psp'nin uzunlugu eklendi.
;
;          Adres degiskeninin Segment word'une eklendi.
;
;          mov      word ptr[adres+2],ax
;
;          Protect konulacak Exe'nin basina dondu.
;
;          mov      ah,19h ; Current disk
;          int      21h
;          sub      al,1
;          jnc      c0

```



```

;
Begin      jmp      cs:adres
CodeSeg    endp
End        Begin

; Bu bölümde 32K 52K file'ı ortak Header
; altında tutulacaktır.
;
StackSeg segment para stack 'stack'
dw 32 dup(?)
StackSeg ends
DataSeg segment para
koru      db 10 dup(?)
exten     db 'exe',0
buf1      db 512 dup(?)
hdl1      dw ?
prtc      db 'c:\tp5\ilogyaz.exe',0
buf2      db 512 dup(?)
hdl2      dw ?
new       db 'New.exe',0
buf3      db 512 dup(?)
hdl3      dw ?
x1        dw ?
x2        dw ?
px1       dw ?
px2       dw ?
ofs       dw ?
sgn       dw ?
DataSeg   ends
CodeSeg   segment para 'code'
Begin     proc far
Assume cs:CodeSeg,ss:StackSeg,ds:DataSeg
;
; Bu bölümde protect konulacak programın ismi
; disaridan giriliyor.
; Bu parametre ESP nin 80H location nina yerlesir.
; 80H de parametrenin uzunlugu ve bir bosluk
; degeri vardir.
; 81H de bosluk 82H den itibaren parametre yazilir.
;
mov si,87h
mov ax,DataSeg
mov es,ax
lea di,koru
sub cx,cx
mov di,byte ptr [si-2]
dec cx
; Okunarin ileri dogru yapilmasi icin.
rdb byte [di-1] ; DS:DI --> PS:DI
mov dx,ax ; mov dx,dece
; Parametrenin sonuna 'exe','e' ekleniyor.
lea si,exten
mov dx,si ; 'exe','e' (3 chars.)
rdb byte
; Protect konulan programin ilk 512 bytes okunuyor.
mov ah,87h ; File aciliyor.
lea dx,koru
mov si,0 ; 1 Byte okunabilir.
int 21h

```



```

; Program Name : Fjoin
; -----
; Bu program iki EXE file'i ortak Header
; altinda birlestirir.
; -----
;
Stackseg segment para stack 'stack'
        dw 32 dup(?)
Stackseg ends
Dataseg segment para
koru     db 30 dup(?)
exten    db '.exe',0
buf1     db 512 dup(?)
hdl1     dw ?
prtct    db 'c:\tp5\flopyaz.exe',0
buf2     db 512 dup(?)
hdl2     dw ?
new      db 'Fnew.exe',0
buf3     db 512 dup(?)
hdl3     dw ?
x1       dw ?
x2       dw ?
px1      dw ?
px2      dw ?
ofs      dw ?
sgm      dw ?
Dataseg  ends
Codeseg  segment para 'code'
Begin    proc far
Assume cs:Codeseg,ss:Stackseg,ds:Dataseg
;
; Bu bolumde protect konulacak programin ismi
; disaridan giriliyor.
; Bu parametre PSP nin 80H location nina yerlesir.
; 80H de parametrenin uzunlugu ve bir bosluk
; degeri vardir.
; 81H de bosluk 82H den itibaren parametre yazilir.
;
        mov     si,82h
        mov     ax,Dataseg
        mov     es,ax
        lea     di,koru
        sub     cx,cx
        mov     cl,byte ptr [si-2]
        dec     cx
        cld
        ; Okumanin ileri dogru yapilmasi icin.
        rep     movsb ; DS:SI ---> ES:DI
        mov     ds,ax ; mov ds,dseg
;
; Parametrenin sonuna .Exe,'0' ekleniyor.
;
        lea     si,exten
        mov     cx,5 ; .Exe,'0' (5 chars.)
        rep     movsb
;
; Protect konulan programin ilk 512 bytes okunuyor.
;
        mov     ah,3dh ; File aciliyor.
        lea     dx,koru
        mov     al,0 ; Sadece okunabilir.
        int     21h

```



```

;      mov      hdl1,ax
;
;      call     read1
;
;      mov      ah,3dh  ; File aciliyor.
;      lea      dx,prct
;      mov      al,0
;      int      21h      ; Sadece okunabilir.
;      mov      hdl2,ax
;
;      call     read2
;
;      mov      ax,word ptr buf2[4]
;      mov      px1,ax
;      mov      ax,word ptr buf2[2]
;      mov      px2,ax
;
;      mov      ax,word ptr buf1[4]
;      mov      x1,ax
;      mov      ax,word ptr buf1[2]
;      mov      x2,ax
;
;      ; Korunulacak programin kalan bytes lari paragraf
;      ; olacak bicimde duzunleniyor.
;      ; Cunku Protect programina giderken IP = 0 olmalı.
;
;      mov      cx,16
;      div      cl
;      sub      cl,ah
;      add      x2,cx
;
;      mov      ax,x2
;      add      ax,word ptr buf2[2]
;      cmp      ax,512
;      jb      done
;      sub      ax,512
;      inc      word ptr buf1[4]
;
;      ; tekrar:
done:
;      mov      dx,word ptr buf2[4]
;      dec      dx
;      add      word ptr buf1[4],dx
;      mov      word ptr buf1[2],ax
;
;      ; Protect konulacak file'in toplam uzunlugunun bulunmasi.
;
;      mov      bx,word ptr buf1[8]
;
;      mov      ax,x1
;      mov      cx,32
;      mul      cx
;      sub      ax,bx
;      mov      dx,x2
;      mov      cl,4
;      shr      dx,cl
;      add      dx,ax
;      sub      bx,bx
;
;      ; Protect konulacak programin CS ve IP degerleri
;      ; memory de saklaniyor.
;      ; Bu degerler Protect programina gonderilecek.
;      ; Adres degiskenine atanacak. Simdilik ofs ve sgm
;      ; degiskenlerinde saklaniyor.
;

```



```

; 4-5 word mov ax,word ptr buf1[14h]; IP degeri...
; gelmekten mov cx,word ptr buf1[16h]; CS degeri...
; mov ofs,ax ; offset = IP
; mov sgmm,cx ; segment = CS
;
; Header'a Protect programinin CS:IP degerleri
; ataniyor. IP:0000 CS:Protect programini Load
; modulunun uzunlugu (parag.).
; Adres degeri hdl3'a atanir. Protect.Exe nin
; calismasi ile Protect.Exe ile programin
; verilecegi adresi bulmak.
;
; NEW.EXE file'inin yaratilmasi.
; Protect.Exe ile protect konulacak herhengibir
; file'in exe'lerinin ortak Header altinda
; birlestirilmesi.
;
; mov ah,3Ch
; lea dx,new
; mov cx,0
; int 21h
; mov hdl3,ax
;
; mov ah,3Dh
; lea dx,new
; mov al,1
; int 21h
; mov hdl3,ax
;
; mov ah,40h
; mov bx,hdl3
; lea dx,buf1
; mov cx,512
; int 21h
;
; mov cx,x1
; dec cx
;
; tekrar:
; push cx
; call read1
; mov ah,40h
; mov bx,hdl3
; lea dx,buf1
; mov cx,512
; int 21h
; pop cx
; loop tekrar
;
; mov ah,3Fh
; mov bx,hdl1
; lea dx,buf1
; mov cx,x2
; int 21h
; mov ah,40h
; mov bx,hdl3
; lea dx,buf1
; mov cx,x2
; int 21h
;
; call read2
;
; Bu page'in 2,3,4 ve 5. byte larina adres
; degiskeni karsilik gelmektedir.
; Bu byte larin 2-3 word'u ----> Offset degerine

```


; 4-5 word'u ----> Segment degerine karsilik
; gelmektedir.

;
mov ax,ofs
mov word ptr buf2[3],ax
mov ax,sgm
mov word ptr buf2[5],ax

;
; Adres degikenindeki bu deger ile Protect.Exe nin
; calismasindan sonra kontrol Protect konulan programa
; verilecektir.

;
mov cx,512
mov ah,40h
mov bx,hdl3
lea dx,buf2
int 21h

;
mov cx,px1
dec cx
dec cx

tkr:
push cx
call read2
mov ah,40h
mov bx,hdl3
lea dx,buf2
mov cx,512
int 21h
pop cx
loop tkr

;
call read2
mov cx,px2
mov ah,40h
mov bx,hdl3
lea dx,buf2
int 21h

;
mov ah,3eh
mov bx,hdl1
int 21h

;
mov ah,3eh
mov bx,hdl2
int 21h

;
mov ah,3eh
mov bx,hdl3
int 21h

;
mov ah,4ch
int 21h

;
Begin endp

;
read1 proc near
mov ah,3fh
mov bx,hdl1
lea dx,buf1
mov cx,512
int 21h
ret
read1 endp


```
;
read2      proc      near 76
            mov      ah,3fh
            mov      bx,hdl2
            lea      dx,buf2
            mov      cx,512
            int      21h
            ret
read2      endp
```

```
;
CodeSeg    ends
            End      Begin
```

Bu dosya, yazılımın olan Copy Protection ile programla gerçekleştirilmiştir. Korunmuş program, kullanıcısı programı ile bu programa konulacak olan bilgileri birleştirir ve bunu bir dosya olarak çalıştıracak olan programdır. Program çalıştırıldığında, DOS 3.86 işletim sisteminin özelliklerinden yararlanılmaktadır.

Projenin sonunda, kişisel bir bilgisayarda abartı yapılmış ya da olan herhangi bir program (protection) ile kopyalanabilmektedir.

SONSÖZ

Bu projedeki amaç, herhangi bir yazılımın izinsiz kopyalanmasını engellemektir. Bu olay, yazılmış olan Copy Protection programıyla gerçekleştirilmiştir. Koruma programı, kullanıcı programı ile bu programa konulacak olan şifreyi birleştiren ve bunu bir bütün olarak çalıştıracak olan programdır. Program oluşturulurken, DOS 3.XX işletim sisteminin özelliklerinden yararlanılmaktadır.

Projenin sonunda, kişisel bir bilgisayarın sabit diskinde yer alan herhangi bir programa (protection) koruma koyulabilmektedir.

5. DUNCAN, R ; ADVANCED MS-DOS
MICROSOFT PRESS, 1986. USA

6. SANCHEZ, J ; ASSEMBLY LANGUAGE TOOLS & TECHNIQUES
FOR THE IBM MICROCOMPUTERS
PRENTICE - HALL, INC, USA

7. ABEL, P ; IBM PC / XT MICROCOMPUTER SOFTWARE
PRENTICE - HALL INTERNATIONAL, INC, USA

8. DISK OPERATING SYSTEMS FOR IBM PC / XT
INTERNATIONAL BUSINESS MACHINES CORPORATION, USA

9. IBM PC / XT TECHNICAL REFERENCE
INTERNATIONAL BUSINESS MACHINES CORPORATION, USA

REFERANSLAR

1. WADLOW, A.T ; MEMORY RESIDENT PROGRAMING ON THE IBM-PC
ADDISON-WESLEY PUB. CO. , 1987, USA
2. NORTON, P ; PROGRAMMER'S GUIDE TO THE IBM-PC
MICROSOFT PRESS , 1985 ,USA
3. HYMAN, M.I ; MEMORY RESIDENT UTILITIES, INTERRUPTS AND
DISK MANAGEMENT WITH MS AND PC DOS
MIS.INC. ,1986, USA
4. GODFREY, J.T ; IBM MICROCOMPUTER ASSEMBLY LANGUAGE
PRENTICE - HALL, INC. , USA
5. DUNCAN, R ; ADVANCED MS-DOS
MICROSOFT PRESS, 1986, USA
6. SANCHEZ; J ; ASSEMBLY LANGUAGE TOOLS & TECHNIQUES
FOR THE IBM MICROCOMPUTERS
PRENTICE - HALL, INC, USA
7. ABEL, P ; IBM PC ASSEMBLER LANGUAGE AND PROGRAMMING
PRENTICE - HALL INTERNATIONAL, INC, USA
8. DISK OPERATING SYSTEM TECHNICAL REFERENCE
INTERNATIONAL BUSINESS MACHINES CORPORATION, 1986
9. IBM PC / AT TECHNICAL REFERENCE
INTERNATIONAL BUSINESS MACHINES CORPORATION, 1984

ÖZGEÇMİŞ

1966 yılında İstanbul'da doğdum. Orta öğretimimin ilk üç yılını İst.Çapa Ortaokulun da, diğer üç yılını İst.Şehremini Lisesi nde bitirdim. 1983 yılında Yıldız Üniversitesi Bilgisayar Bilimleri Mühendisliğine girdim ve 1987 yılında bu bölümden mezun oldum. Halen bu bölümde Arş.Grv. olarak görev yapmakta ve yüksek öğrenimime devam etmekteyim.

