📘 Lecture 7 — Kernel, PowerShell, BIOS, Secure Boot, BDS & RDP

---

1. Kernel

What it is

The **kernel** is the core part of an operating system. It mediates between hardware (CPU, memory, disk, network, devices) and user programs, providing essential services such as scheduling, memory management, I/O and device control.

How it works (high level)

- Programs request services via **system calls** (e.g., open file, allocate memory).
- The kernel performs low-level tasks (talks to drivers, schedules CPU time, manages RAM) and returns results to programs.
- It runs in **kernel space** (privileged) while normal programs run in **user space** (restricted).

Common kernel functions

- **Process scheduling** (who runs when).
- **Memory management** (allocating RAM, swapping).
- **Device & driver management** (loading drivers, I/O).
- **System call handling** (interface for apps).
- **Security & isolation** (privileges, permissions).

Types of kernels

- **Monolithic kernel** — most services run in kernel space (Linux). Pros: fast (fewer context switches). Cons: large codebase, harder to isolate faults.
- **Microkernel** — minimal core, many services in user space. Pros: modular, easier to isolate faults. Cons: potential performance overhead.
- **Hybrid kernel** — mix of both (Windows uses hybrid approach).

Kernel & security

- **Kernel vulnerabilities** are high-impact (rootkits, privilege escalation).
- **BSOD (Blue Screen of Death)** often occurs when a kernel-level driver crashes or corrupts memory — common causes: bad drivers, hardware faults, or kernel exploitation.
- **Driver impact:** a buggy or malicious driver can crash the kernel or allow attackers to run code in kernel space.

---

2. PowerShell

What it is

PowerShell is Microsoft's powerful shell and scripting environment (built on .NET). It provides cmdlets and scripting to manage Windows (files, services, registry, processes, network).

How it works & capabilities

- Can execute commands and scripts (.ps1) to manage system state (create/delete files, start/stop services, query registry).
- Integrates with Windows APIs and can call native commands.
- Supports remote management (WinRM / PowerShell Remoting).

Common PowerShell examples

- List processes:
- Get-Process
- List services:
- Get-Service
- Download a file:
- Invoke-WebRequest -Uri "https://example.com/file" -OutFile "C:\temp\file.exe"
- Delete a file (use with extreme caution):
- Remove-Item -Path "C:\temp\malware.exe" -Force

Hibernation file & pagefile (PowerShell context)

- **hiberfil.sys** — Windows hibernation file (stores RAM to disk on hibernate). Disable hibernation (this deletes hiberfil.sys):
- powercfg /hibernate off
- **pagefile.sys** — Windows pagefile (virtual memory). Pagefile settings are managed via System → Advanced → Performance → Advanced → Virtual memory or via WMI/PowerShell for advanced admins.

Security notes & risks

- PowerShell is **commonly abused by attackers** for post-exploitation (downloading payloads, privilege escalation, living-off-the-land).
- Defenders should enable **constrained language mode**, enable **PowerShell logging** (Module Logging, Script Block Logging, Transcription) and set appropriate **ExecutionPolicy** (e.g., AllSigned) to restrict unsigned scripts.
- Example to set execution policy (requires admin):
- Set-ExecutionPolicy AllSigned -Scope LocalMachine

---

3. BIOS (Basic Input/Output System) & UEFI

What BIOS/UEFI is

- Firmware on the motherboard that **initializes hardware** at boot (POST — Power On Self Test) and loads the OS bootloader into memory.

How to check BIOS mode in Windows

- Press Win + R, type:
- msinfo32

In **System Information**, look for **BIOS Mode** (shows UEFI or Legacy).

BIOS vs UEFI (types)

- **Legacy BIOS (MBR)**
  - Older boot method; uses **MBR (Master Boot Record)** partitioning.
  - Limits: 2 TB disks, 4 primary partitions.

- **UEFI (GPT)**
    - Modern firmware with GUI options, supports **Secure Boot**, fast boot, NVMe, GPT partition style.
    - Advantages: supports disks >2 TB, more flexible partitioning, improved security features.

Role of BIOS/UEFI in security

- UEFI provides features like **Secure Boot** and firmware-level protections. Firmware compromises (UEFI rootkits) are stealthy and very persistent.

---

4. Secure Boot

What it is

Secure Boot is a UEFI feature that ensures only trusted, digitally-signed bootloaders and drivers run during the boot process.

How it works

- UEFI has a set of trusted **public keys**.
- At boot, the firmware verifies the **digital signature** of the bootloader and essential drivers.
- If the signature isn't valid (unsigned or tampered), the firmware blocks execution — preventing unsigned or tampered OS loaders from running.

Why it's important

- Protects against **bootkits/rootkits** that load before antivirus starts.
- If malware manages to insert itself into the boot chain, Secure Boot helps prevent it from running.

Practical notes & caveats

- **Cracked software** or rogue boot components often lack valid digital signatures — Secure Boot can block them.
- **Antivirus timing:** Antivirus starts after the OS is loaded; if malware runs before OS load (bootkit), it can persist and disable AV — Secure Boot is a preventative control.

How to check/enable Secure Boot

- In UEFI/BIOS settings (access during boot, e.g., F2/Del) — check the Secure Boot option.
- In Windows msinfo32 you can also see **Secure Boot State**.

---

5. BDS — Behavior Detection System (Behavior-based Detection)

What it is

A **Behavior Detection System (BDS)** monitors runtime **behavior** of applications/processes (instead of just matching signatures) to detect malicious or anomalous activity.

How it works

- Observes actions like unusual process behavior, network connections, file modifications, API calls, attempts to escalate privileges.
- Uses heuristics, rules, and often **machine learning** to detect suspicious patterns (e.g., a browser suddenly trying to open RDP connections).
- Often part of modern endpoint protection: EDR (Endpoint Detection & Response) systems include behavior analysis.

Why it's used

- Signature-based AV misses **zero-day** or polymorphic malware. Behavior detection can catch never-before-seen threats by their actions.
- Detects fileless attacks, script-based abuses, unusual lateral-movement behavior.

Examples of what it can detect

- A text editor or browser process spawning cmd.exe and making outbound network connections on uncommon ports.
- A process modifying many files rapidly (possible ransomware).
- Suspicious use of system utilities (PowerShell, wmic, schtasks) to achieve persistence.

Practical defender actions

- Use EDR/BDS tools that provide visibility and response (isolate host, kill process, collect forensic data).
- Tune rules to reduce false positives; keep telemetry and logs centralized for analysis.

---

6. RDP (Remote Desktop Protocol)

What it is

RDP is Microsoft's protocol for remote graphical desktop access (native client mstsc.exe). Default port: **3389 (TCP)**.

How it works (basic)

- Client connects to server's RDP port, establishes authentication (can use Network Level Authentication/NLA), then remotes the display, keyboard, and mouse.
- Data can be encrypted (TLS/RDP encryption).

Security risks

- **Exposed RDP** on the Internet is heavily targeted for brute-force and credential-stuffing attacks.
- Vulnerabilities in the RDP service have historically allowed remote code execution (patch quickly).
- Attackers use compromised credentials or RDP exploits to gain full access and deploy ransomware.

Hardening & best practices

- **Disable direct Internet exposure** of RDP; use VPN or jump host.
- Enable **Network Level Authentication (NLA)** to require client authentication before establishing a session.
- Use **strong passwords** and **multi-factor authentication** (MFA).
- Change default port (security by obscurity — slows automated scans but not a real defense).
- Limit RDP access using firewall rules (allow only specific IPs).

- Keep RDP patched; monitor logs for failed logins and unusual connections.
- Consider solutions like **Just-In-Time** access (Azure) or **Remote Desktop Gateway** for secure access.

Quick Practical Commands & Steps (safe reminders)

- **Check BIOS mode and Secure Boot in Windows:**
    1. Win + R → msinfo32 → Look for **BIOS Mode** and **Secure Boot State**.
- **Disable/Enable Hibernation (delete/restore hiberfil.sys):**
- powercfg /hibernate off # disable and delete hiberfil.sys
- powercfg /hibernate on # enable hibernation again
- **Set PowerShell execution policy (restrict scripts):**
- Set-ExecutionPolicy AllSigned -Scope LocalMachine
- **Enable PowerShell logging (admins):** enable Module Logging and Script Block Logging via Group Policy or registry — helps detect malicious scripts.
- **Check RDP port & status:**
    - Check listening port:
    - netstat -ano | findstr 3389
    - Test remote connection:
    - Test-NetConnection -ComputerName <ip-or-host> -Port 3389

Summary Table (compact)

Topic	What	Why important	Security notes
Kernel	OS core managing hardware & processes	Central to system function	Kernel bugs = critical; drivers can crash system (BSOD)

Topic	What	Why important	Security notes
PowerShell	Powerful scripting shell	Admin automation & remote management	Frequently abused; enable logging & restrict execution
BIOS / UEFI	Firmware that boots the OS	Initializes hardware; UEFI supports Secure Boot	Firmware attacks are persistent; check mode with msinfo32
Secure Boot	UEFI verification of signed boot components	Prevents bootkits/rootkits	Blocks unsigned bootloaders; can stop some malware
BDS	Behavior-based detection engine	Detects zero-days & fileless attacks	Use EDR with behavioral rules; tune to reduce false positives
RDP	Remote desktop protocol (port 3389)	Remote management	Harden: VPN, NLA, MFA, restrict access & patch regularly

✅ ⭐ Summary of Lecture 7 — Kernel, PowerShell, BIOS, Secure Boot, BDS & RDP

1. Kernel

- The kernel is the **core of the operating system**.
- Manages **CPU, memory, devices, processes, and system calls**.
- Runs in **kernel mode** (full permissions) while apps run in **user mode**.
- Types:
  - **Monolithic (Linux)** – faster, large.
  - **Microkernel (QNX/Minix)** – modular, more secure, slower.
  - **Hybrid (Windows)** – mix of both.
- Kernel issues can cause **BSOD** and vulnerabilities like **privilege escalation**.

2. PowerShell

- Command-line + scripting environment by Microsoft.
- Manages **files, processes, registry, network**.
- Can run commands, scripts, and automate complex tasks.
- Attackers often abuse it—requires **logging & execution policy control**.

3. BIOS & UEFI

- Firmware that initializes hardware and loads the OS.
- **Legacy BIOS (MBR)** is old; supports <2TB disks.
- **UEFI (GPT)** is modern; supports secure boot, large disks, better security.
- Check BIOS/UEFI using **msinfo32**.

4. Secure Boot

- Ensures only **digitally signed** bootloaders & drivers are executed.
- Prevents **bootkits, rootkits, and unauthorized OS loaders**.
- Part of modern UEFI systems.

5. BDS (Behavior Detection System)

- Detects malicious behavior based on **actions**, not just signatures.
- Used in **EDR tools**.
- Detects zero-days, fileless malware, unusual process activities.

6. RDP (Remote Desktop Protocol)

- Enables remote Windows access (default port **3389**).
- Useful for remote management but heavily targeted.
- Must be hardened using:
  - **VPN**
  - **NLA**
  - **MFA**
  - **Firewall filtering**
  - **Patching**

✅ ⭐ Conclusion

This chapter explains the **deep internal mechanisms of system security**.
You learned:

- **How OS foundations work (Kernel, Drivers, Firmware).**
- **How administrative tools (PowerShell, RDP) enable automation & remote access.**
- **How security layers (Secure Boot & BDS) protect against advanced cyber threats.**
- **How system-level vulnerabilities can lead to major attacks like bootkits, rootkits, RDP ransomware infections.**

This knowledge is extremely valuable in **ethical hacking, system administration, SOC, digital forensics, and cyber defense**.

✅ ⭐ Detailed Mindmap (Text Version)

LECTURE 7 — Kernel, PowerShell, BIOS, Secure Boot, BDS, RDP

```
|
├── 1. Kernel
|     ├── Definition: OS core
|     ├── Runs in kernel mode
|     ├── Functions:
|     |     ├── Process scheduling
|     |     ├── Memory management
|     |     ├── Device/driver control
|     |     ├── System calls handling
|     ├── Kernel Types:
|     |     ├── Monolithic (Linux)
|     |     ├── Microkernel
|     |     ├── Hybrid (Windows)
|     ├── Security:
|           ├── Kernel exploits
|           ├── Privilege escalation
|           ├── BSOD from driver failure
|
├── 2. PowerShell
|     ├── Advanced shell by Microsoft
|     ├── Cmdlets (Get-Process, Get-Service)
|     ├── Scripting (.ps1)
|     ├── Remote management (WinRM)
```

```
|     ├── Access to APIs, registry, network
|     ├── Security:
|         ├── ExecutionPolicy
|         ├── Logging (ScriptBlock, Module)
|         ├── Used in attacks (LOLBIN)
|
├── 3. BIOS / UEFI
|     ├── Firmware that boots the OS
|     ├── Legacy BIOS (MBR)
|     ├── UEFI (GPT)
|     ├── Check using msinfo32
|     ├── Security:
|         ├── Firmware attacks
|         ├── Secure Boot support
|
├── 4. Secure Boot
|     ├── Verifies digital signatures
|     ├── Blocks unsigned bootloaders
|     ├── Prevents rootkits/bootkits
|     ├── Enabled via UEFI settings
|
├── 5. BDS (Behavior Detection System)
|     ├── Detects behavior patterns
|     ├── Finds zero-days
|     ├── Detects fileless malware
```

```
|     ├── Part of EDR platforms
|
└── 6. RDP (Remote Desktop Protocol)
    ├── Remote access (port 3389)
    ├── Risk: brute-force, exploits
    ├── Hardening:
    |     ├── NLA
    |     ├── MFA
    |     ├── VPN
    |     ├── Firewall rules
    |     ├── Patch regularly
```

✅ ⭐ Complete Interview Questions + Best Answers

🔹 Kernel — Interview Questions

1. What is a Kernel?

The kernel is the **core component of the OS**.
It manages hardware resources like CPU, memory, and devices.
Programs communicate with hardware through **system calls**, and these are handled by the kernel.

---

2. What are Kernel Modes?

There are two modes:

1. **Kernel Mode** – full access to hardware (OS + drivers run here).
2. **User Mode** – restricted access (applications run here).

This separation prevents apps from damaging the system.

---

3. Difference between Monolithic and Microkernel?

Monolithic	Microkernel
Entire OS services run in kernel space	Only essential services run in kernel space
Faster performance	More secure & stable
Harder to isolate failures	Modular

Linux = Monolithic
Windows = Hybrid (mix of both)

---

4. What causes a BSOD?

A BSOD occurs due to a **critical kernel-level error**, usually from:

- Faulty drivers
- Memory corruption
- Kernel exploit
- Hardware failure

🔹 PowerShell — Interview Questions

5. What is PowerShell?

PowerShell is a **command-line + scripting platform** by Microsoft used to automate system tasks, manage Windows services, registry, files, and network.

---

6. Why do attackers use PowerShell?

Because:

- It is **built into Windows**
- Can run **in memory (fileless)**
- Can download files
- Can escalate privileges

- Bypasses many security tools

It's a popular **post-exploitation** tool.

---

7. How do you secure PowerShell?

- Enable **Script Block Logging**
- Enable **Module Logging**
- Set execution policy to **AllSigned**
- Restrict PowerShell v2 (older, less secure)
- Use **Constrained Language Mode**

🔹 **BIOS / UEFI — Interview Questions**

8. What is the difference between BIOS and UEFI?

BIOS is older, supports MBR, <2TB disks.
UEFI is modern, supports GPT, Secure Boot, NVMe, GUI, and better security.

---

9. What is POST?

POST = **Power On Self Test**
Firmware checks CPU, RAM, hardware before booting the OS.

🔹 **Secure Boot — Interview Questions**

10. What is Secure Boot?

Secure Boot ensures only **trusted, digitally signed** bootloaders & drivers load.
It protects against **bootkits and rootkits**.

---

11. How does Secure Boot work?

- UEFI stores **trusted certificate keys**
- Boot components are verified

- If signature is invalid → boot process stops

🔹 BDS — Interview Questions

12. What is a Behavior Detection System?

A security system that detects malware based on **actions**, not signatures.
Used in **EDR tools** to detect:

- Zero-days
- Ransomware
- Fileless attacks
- Abnormal process behaviors

---

13. Why is behavior detection important?

Signature-based antivirus cannot detect new or polymorphic malware.
Behavioral detection catches malware based on **what it does**, not what it looks like.

🔹 RDP — Interview Questions

14. What is RDP?

Remote Desktop Protocol allows remote Windows access using **port 3389**.

---

15. What are risks of RDP?

- Brute-force attacks
- Credential theft
- RDP exploits
- Ransomware spreading
- Unauthorized access

---

16. How to secure RDP?

- Enable **NLA**
- Use **VPN**
- Use **MFA**
- Restrict IPs via **firewall**
- Keep patched
- Monitor login attempts

---

17. What is NLA in RDP?

Network Level Authentication requires the user to authenticate **before** the RDP session is created.
This prevents many brute-force and exploit attempts.