Slide 1: Title Page / 标题页

(0:00 - 0:45)

[CN] 尊敬的主席，委员会成员，大家早上好。我是袁伟泽。 今天我汇报的题目是《服务器与类 Unix 系统在智能家居传感器控制中的应用》。 我的导师是 Miroslav Husák 教授。

[EN] Honorable Chairman, members of the committee, good morning. My name is Yuan Weize. My thesis topic is **"Application of Servers and Unix-like Systems for Sensor Control in Smart Homes."** My supervisor is Professor Miroslav Husák.

Slide 2: The Smart Home Dilemma / 智能家居的困境

(0:45 - 1:30)

[CN] 现有的商业智能家居面临三大痛点： 第一，**隐私泄露**。数据上传云端，不安全。 第二，**厂商锁定**。品牌之间互不兼容。 第三，**断网瘫痪**。没有互联网就无法工作。 我的目标是构建一个 **100% 本地化控制** 的系统，把数据主权还给用户。

[EN] Current commercial smart homes face three major issues:

1. **Privacy Risks**: Data is stored on the cloud.
2. **Vendor Lock-in**: Devices are incompatible.
3. **Internet Dependency**: No internet means no control. My goal is to build a system with **100% Local Control**, giving **Data Sovereignty** back to the user.

Slide 3: System Architecture / 系统架构

(1:30 - 2:15)

[CN] 这是我的系统架构，采用了标准的**三层设计**： 最顶层是**应用层**（Home Assistant）， 负责用户交互。 中间是**网络层**（MQTT Broker）， 负责消息路由。 最底层是**传输层**（ESP32 传感器）， 负责硬件采集。 这种分层设计确保了系统的可扩展性和维护性。

[EN] This is the system architecture, featuring a standard **Three-Tier Design**:

- The top is the **Application Layer** (Home Assistant) for user interaction.
- The middle is the **Network Layer** (MQTT Broker) for message routing.
- The bottom is the **Transport Layer** (ESP32 sensors) for hardware data collection. This design ensures **scalability** and **maintainability**.

Slide 4: Security Architecture / 安全架构

(2:15 - 3:00)

[CN] 这是本项目的核心亮点——**安全架构**。 大多数 DIY 项目只用简单的密码，极不安全。我构建了一套完整的 **PKI (公钥基础设施)**。 在固件中，我**显式强制 (Explicitly Enforced)** 使用 **ECDSA P-256** 和 **X25519** 加密算法。 为什么要这样做？是为了防止底层库更新带来的潜在风险。通过"硬编码"这些高标准，我实现了 **"Deploy and Forget" (部署即忘)** 的稳定性——无论未来如何升级，我的安全标准永远最高，黑客无法进行降级攻击。

[EN] This is the core highlight: the **Security Architecture**. Most DIY projects rely on weak passwords. I built a full **PKI (Public Key Infrastructure)**. I **Explicitly Enforced** strong cipher suites like **ECDSA P-256** and **X25519** in the firmware. **Why?** To prevent any potential downgrade attacks if the underlying libraries change defaults. By hard-coding these standards, I achieved a **"Deploy and Forget"** stability—ensuring the system remains secure long-term.

Slide 5: MQTT Architecture / MQTT 架构设计

(3:00 - 3:45)

[CN] 为了高效管理数据，我设计了一个**层级化**的 MQTT 架构。 大家看图：左边是作为发布者的 ESP32，中间是加密的 EMQX 代理，右边是作为订阅者的 Home Assistant。 我定义了清晰的 **主题结构 (Topic Structure)**， 比如 `home/bedroom/sensor1`。这种设计让系统不仅能容纳几个设备，甚至可以轻松扩展到上百个传感器，互不干扰。

[EN] To manage data efficiently, I designed a **Hierarchical MQTT Architecture**. As shown in the diagram:

- On the **Left**: Publishers (ESP32 sensors).
- In the **Middle**: EMQX Broker (Encrypted routing).
- On the **Right**: Subscribers (Home Assistant). I defined a clear **Topic Structure**, such as `home/bedroom/sensor1`. This allows the system to scale from a few devices to hundreds **without conflict**.

Slide 6: Hardware Integration / 硬件集成

(3:45 - 4:30)

[CN] 在硬件端，我选择了 **ESP32** 芯片。 为了提高效率，我使用了 **ESPhome** 框架。它是基于 **YAML** 配置文件的。 相比于复杂的 C++， YAML 非常直观， 而且支持 **OTA (在线升级)**。 这让我可以在不拆卸外壳的情况下，远程调整传感器参数，非常适合快速迭代。

[EN] For the hardware, I chose the **ESP32** chip. To improve efficiency, I used the **ESPhome framework**, which is based on **YAML** configuration. Compared to complex C++, YAML is intuitive and supports **OTA (Over-the-Air) updates**. This allows me to tune sensor parameters **remotely** without opening the device case.

Slide 7: Fall Detection (Answer to Opponent) / 跌倒检测

(4:30 - 5:45) *(重点：回答 Jan Koller)*

[CN] 这是我的跌倒检测模块 (MPU6050)。 在这里，我想深入回答 **Jan Koller 博士**提出的关键问题：*“为什么跌倒检测需要结合加速度（Accel）和角速度（Gyro）？”*

[EN] This is the fall detection module using the MPU6050 sensor. I would like to address the critical question raised by the opponent, **Dr. Jan Koller**: *"Why combine Acceleration and Angular Velocity for fall detection?"*

[CN] 我的回答基于两种技术路线的对比： 第一，对于本项目采用的**基于阈值**的算法，文献表明仅用加速度计容易误报（比如把跳跃当成跌倒）。引入角速度可以将**特异性（Specificity）**从 82% 提升到 **96%**。它能区分“高冲击力”和“姿态翻转”。

第二，我知道如果有算力支持，使用**深度学习（CNN）也可以仅用加速度计实现高精度。 但是，作为工程师，我必须做权衡。为了在 ESP32 上保持低功耗和高效率**，我没有运行沉重的 AI 模型。 因此，“传感器融合”是当前架构下**最稳健**的工程选择。

[EN] My answer is based on comparing two approaches: First, for the **Threshold-based algorithm** I used, literature shows that an Accelerometer alone leads to false alarms (like jumping). Adding **Angular Velocity (Gyroscope)** improves **Specificity** from 82% to over **96%**. It distinguishes "High Impact" from actual "Rotational Falls."

Second, I am aware that with **Deep Learning (CNNs)**, an accelerometer alone *could* be sufficient. **However**, as an engineer, I made a trade-off. To keep the ESP32 efficient with **Low Power Consumption**, I avoided heavy AI models. Therefore, the **Sensor Fusion** approach is the most **robust engineering choice** for this architecture.

Slide 8: Node-RED Logic Flow / 逻辑流程图

(5:45 - 6:30)

[CN] 除了硬件，我还设计了软件自动化逻辑。 这是 **Node-RED** 的实际截图。不仅仅是简单的“如果-那么”，我设计了一个**“状态机 (State Machine)”**：

1. 检测传感器状态（跌倒）。
2. 检查系统状态（是否布防）。
3. 发送 iOS 推送通知。 这种可视化流程让后期的维护变得非常直观。

[EN] Beyond hardware, I designed the software automation logic. This screenshot shows my actual **Node-RED** flow. It acts as a **State Machine**:

1. Detects the sensor state (Fall Detected).
2. Checks system status (Is it armed?).
3. Sends a push notification to iOS. This visual flow makes future maintenance extremely intuitive.

Slide 9: Scalability & Performance / 扩展性与性能测试

(6:30 - 7:15)

[CN] 作为工程师，必须验证系统极限。这是我的**压力测试**结果。我用 Python 脚本模拟了 **50 个**并发客户端。结果显示：在 ESXi 服务器支持下，CPU 占用率很低，且消息延迟保持在 **10 毫秒** 以内。这证明系统完全能支撑复杂的智能家庭环境。

[EN] As an engineer, I must verify system limits. This chart shows my **Scalability Test**. I simulated **50 concurrent MQTT clients**. The result: Backed by the ESXi server, CPU usage remained low, and latency stayed under **10 milliseconds**. This proves the system can easily support a complex smart home environment.

Slide 10: Platform Analysis / 平台对比分析

(7:15 - 8:00)

[CN] 接下来是对比分析。请看这张矩阵。我们必须承认一个**战略取舍 (Strategic Trade-off)**：商业平台胜在"即插即用"。但是，我的开源方案在**数据安全**、**隐私控制**和**离线功能**上完胜。最重要的是**厂商独立性**。我的系统永远不会被"厂商锁定"，无论外部商业环境如何变化，系统都长期稳定。

[EN] Next, I compared my system against commercial platforms. Please look at this matrix. We must acknowledge a **Strategic Trade-off**: While commercial platforms are "Plug & Play," My open-source solution wins decisively in **Data Security**, **Privacy Control**, and **Offline Functionality**. Most importantly, it ensures **Vendor Independence**. Unlike proprietary ecosystems, this approach guarantees long-term stability regardless of external business changes.

Slide 11: Contributions & Conclusion / 贡献与结语

(8:00 - 9:00)

[CN] 最后，我想总结本论文的贡献并展望未来。我的核心贡献在于建立了一个完整的闭环系统：从固件工程到安全架构，再到系统集成，每一层都经过了严格设计。

展望未来，我的技术路线图非常清晰：首先，我计划在端侧引入 **TFLite Micro** 以增强智能，并完善 **3D 打印外壳**。最后，我将引入 **5G 模块** 和 **Kubernetes**，以支持多家庭管理和更长期的可靠性。以上就是我的汇报。感谢各位的聆听，我准备好回答问题了。

[EN] Finally, I would like to summarize my contributions and future roadmap. My core contribution is building a complete, closed-loop system: from **Firmware Engineering** and **Security Architecture** to **System Integration**. Every layer is rigorously designed.

Looking ahead, my technical roadmap is clear: First, I plan to implement **On-device AI using TFLite Micro** and refine the hardware with **3D-printed enclosures**. Finally, I aim to introduce **5G modules** and **Kubernetes** deployment to support multi-home management and long-term reliability.

That concludes my presentation. Thank you for listening. I am now ready for your questions.

💡 最后的**临场建议**

1. **Slide 10 技巧**：
   - 讲到 "Strategic Trade-off" 时，可以稍微摊开手，表示这是一种权衡，但你的选择（隐私/自由）是更长远、更高级的。
2. **Slide 11 结尾**：
   - 讲完最后一句 "I am now ready for your questions" 后，**保持微笑，眼神自信地看向主席（Chairman Müller）或对手（Jan Koller）**，等待他们的发问。
3. **联系方式**：
   - 虽然不用念出来的，但你的 PPT 上留了 GitHub 和邮箱，这很好，显示了你的开源精神。如果有人问代码在哪里，可以直接指一下屏幕下方。

💡 **临场发挥建议 (Final On-Stage Strategy)**

1. 眼神与肢体语言 (The "Power Move")

- **开场 (Slide 1)**：不要看屏幕。看着主席（Müller）和委员会，自信微笑。站稳，不要晃动。
- **关键时刻 (Slide 7 - 回答 Koller)**：
  - 当读到 *"I would like to address the critical question..."* 时，**身体转向 Jan Koller**（如果他在现场）。
  - **眼神锁定**：看着他的眼睛说出 *"Specificity increased from 82% to 96%"*。这表示你非常重视他的问题，且答案确凿。
- **讲图表 (Slide 3/5/8/9)**：
  - 使用激光笔或手势指向图表的核心部分（例如指向 MQTT Broker 的中间位置）。
  - 不要背对观众读屏幕，**侧身**站立。

2. 语速控制 (Pacing)

- **慢下来**：在讲 **Slide 4 (Security)** 时，特别是读到 *"ECDSA P-256"* 和 *"Explicitly Enforced"* 时，语速放慢，加重语气。这能体现你的专业性（Professionalism）。
- **快一点**：在 **Slide 2 (Dilemma)** 和 **Slide 10 (Comparison)** 时，语速可以稍快，因为这些是背景和常识，不需要听众费脑力。

3. 应对突发状况 (Emergency Handling)

- **忘词了**：不要慌，看一眼 PPT 的标题，然后用最简单的英语说核心意思（比如："Basically, local control is safer."）。
- **被打断**：如果老师中途提问（虽然 SZZ 通常是最后问），停下来，听完，回答，然后说 *"Shall I continue?"*。

❓ 高频预测问答 (Potential Q&A) - 中英对照版

Category 1: Reliability & Hardware (可靠性与硬件)

(Müller 主席或 Kyncl 副主席最爱问)

Q1: 断网或死机怎么办？ [CN] 问题: 如果 Wi-Fi 断了或者服务器宕机了，系统会发生什么？ **[EN] Question**: What happens if the Wi-Fi or the Server goes down?

- **[CN] 回答逻辑**:
    1. Wi-Fi 断：ESP32 会自动重连。目前数据会丢失，未来计划加 SD 卡缓存。
    2. 服务器死：因为用了 ESXi，可以用快照几分钟恢复。且硬件开关（如墙壁开关）是独立的，不受影响。
- **[EN] Answer**: "If **Wi-Fi fails**, the ESP32 will attempt to reconnect automatically. Currently, data is lost during downtime, but I plan to add a local buffer (SD card) in the future. If the **Server fails**, since I use **ESXi**, I can restore the system from a snapshot in minutes. Also, physical wall switches work independently."

Q2: 供电和电池续航？ [CN] 问题: 你是如何给 ESP32 供电的？电池能用多久？ **[EN] Question**: How do you power the ESP32 sensors? What is the battery life?

- **[CN] 回答逻辑**: 目前是用 USB 线供电（为了稳定）。如果要用电池，ESP32 耗电较大，必须用"深度睡眠模式"，可以将续航延长到几周，但目前原型机是插电的。
- **[EN] Answer**: "Currently, the prototype is **USB-powered** for stability. For battery operation, ESP32 is power-hungry. I plan to implement **Deep Sleep mode**, which could extend battery life to several weeks. But for now, it uses a cable."

Q3: 为什么选 MQTT 而不是 HTTP？ [CN] 问题: 为什么在通信协议上选择 MQTT 而不是 HTTP？ **[EN] Question**: Why did you use MQTT instead of HTTP?

- **[CN] 回答逻辑**:
    1. 效率：MQTT 是轻量级的，协议头很小，省带宽。
    2. **实时性**：MQTT 支持"推送(Push)"，而 HTTP 需要设备不断"轮询(Polling)"，轮询非常浪费 CPU 和电量。
- **[EN] Answer**: "Two reasons: **Efficiency** and **Real-time** performance. MQTT is lightweight and uses less bandwidth. Also, it supports **'Push'** notifications immediately, whereas HTTP requires **'Polling'** (checking constantly), which wastes CPU and energy."

Category 2: Physics & Sensors (物理与传感器)

(对手 Jan Koller 可能会追问细节)

Q4: 传感器校准了吗？ [CN] 问题: 你对 MPU6050 传感器进行校准了吗？ **[EN] Question**: Did you calibrate the MPU6050 sensor?

- **[CN] 回答逻辑**: 做了静态校准。我把它放在平整的桌面上，测量了"零重力偏差(Zero-g offset)"，然后在 YAML 配置文件里减去了这个偏差值。

- **[EN] Answer**: "Yes, I performed a **static calibration**. I placed the sensor on a flat table to measure the **'Zero-g' offset**, and then I subtracted this value in the YAML configuration code to correct the readings."

Q5: 温度会影响读数吗？ [CN] 问题: 温度变化会影响 MEMS 传感器的准确性吗？ **[EN] Question**: Does temperature affect the sensor readings?

- **[CN] 回答逻辑**: 是的，MEMS 都有温漂。但在家庭环境（20-25 度）下，这个误差对跌倒检测来说可以忽略不计。如果是工业级应用，我会加温度补偿算法。
- **[EN] Answer**: "Yes, MEMS sensors have **temperature drift**. However, in a home environment (around 20 to 25 degrees), the error is **negligible** for fall detection. For industrial use, I would implement a temperature compensation algorithm."

Category 3: Security (安全)

(Bauer 或 Boura 可能问)

Q6: 物理安全（设备被偷）？ [CN] 问题: 如果有人把你的 ESP32 偷走了，他们能把证书（私钥）提取出来吗？ **[EN] Question**: What if someone steals your ESP32 device? Can they extract the certificate?

- **[CN] 回答逻辑**: 目前的原型机是有风险的（存在 Flash 里）。**解决方案**：在商业版中，我会启用 ESP32 的 **Flash 加密** 和 **安全启动 (Secure Boot)**，这样即使拆下芯片也读不出密钥。
- **[EN] Answer**: "That is a valid risk. Currently, the key is stored in flash memory. **Solution**: In a commercial version, I would enable **Flash Encryption** and **Secure Boot** (which ESP32 supports). This ensures the key cannot be read even if the chip is physically removed."

Q7: 为什么不用现成的 Matter 或 Zigbee？ [CN] 问题: 为什么要自己搞一套 Wi-Fi + MQTT？为什么不用 Matter 或 Zigbee？ **[EN] Question**: Why not use Matter or Zigbee standards?

- **[CN] 回答逻辑**:
  1. Zigbee 需要额外的网关硬件。
  2. Matter 虽然好但实现太复杂。
  3. Wi-Fi + MQTT 让我能**完全掌控安全层 (mTLS)**，这正是我论文想要研究的核心——如何构建一个自有的安全架构。
- **[EN] Answer**: "Zigbee requires a special hardware gateway. Matter is great but complex to implement from scratch. Using **Wi-Fi + MQTT** allows me to have **full control over the Security Layer (mTLS)**, which was the main research goal of my thesis."

💡 临场应对小抄 (Cheat Sheet)

- 听不**懂问题时**:
  - "Could you please repeat the question?" (请您重复一遍好吗？)
  - "If I understand correctly, you are asking about..." (如果我理解没错的话，您问的是...)
- 不**会回答时**:

- "That is an interesting angle. I haven't tested that specifically, but in theory..." (这个角度很有趣。我没专门测过，但理论上...)
- "I will look into that for my future improvements." (我会把它作为未来的改进点。)