1. A summary of the article on "Reflections on Trusting Trust" by Ken Thompson

The moral of the whole article is that you can't trust the code that is not created by yourself which means that code that you use that only can be trusted if it is from the genuine work else it leads to a serious problem of the trojan horse problem, as example given in the article itself.

The explanation of Ken Thompson is divided into three stages where he takes about the fun programming exercise of self generating code and then code flow of the compiler and how the compiler is vulnerable to the malicious code.

Let's start with the stage - 1

In the stage-1 Ken Thompson talks about the fun exercises they used to do during college and the problem is to write a source program that, when compiled and executed, will produce as output an exact copy of its source. By this he shows that the program can be written by another program and this program contains an arbitrary amount of excess baggage that will be reproduced along with the main algorithm.

stage -2 and 3

In the stage-2 Ken Thompson talks about the compiler in the c programming language containing a set of the escape characters /n,/t and many more. He used the regenerating code from the stage 1 and added /v to the compiler by doing this he showed no amount of source-level verification or scrutiny will protect you from using untrusted code. In demonstrating the possibility of this kind of attack, he picked on the C compiler. He could have picked on any program-handling program such as an assembler, a loader, or even hardware microcode. As the level of the program gets lower, these bugs will be harder and harder to detect. A well-installed microcode bug will be almost impossible to detect.

2. A Summary of how the Internet (including its security) was designed by watching Youtube talk by David D. Clark

In this video Sir David D Clark was promoting his book How the internet design where he has talked about several chapters about how he included it and what is history and why it is important in the Internet design. This is the special part which i liked was when he went to the biggest ISP of the USA and about implementing secure protocol and in reply he said no then he realized the competitive interface the specification of the Internet Protocol which is a cut point in the stack is a money insulator.

In this video he also talked about the phase where the internet was moving from research and private to commercial where he talked about BGP protocol. Where he also talked about the security “security when you begin to break the problem into parts you realize that security is not a dimension along which you're optimizing and here's perfection out there it doesn't work security is a multi-dimensional space in which different objectives actually compete with each other and different stakeholders compete with each other and to build a successfully secure internet is to say you can build a compromise in which all of the actors will will allow your solution to survive.

He also talked about the availability where people focused on the integrity and other where they forgot to solve the problem with availability and cryptography can’t solve the problem of the availability.

He also good things about the network security is he broke the problem in the book into four parts which is people who trust each other trying to communicate and there's a third party who's trying to interfere steal modify block that's the classic problem of information security the second problem is “I connect to you and you attack me you know”, I go to a website and it downloads malware or - you know I download email and it turns out to be spam or have a malicious attachment the third problem is the mechanisms of the the internet that I'm talking about are themselves broken and most of the mechanisms in the Internet today are broken and the fourth one is denial of service attacks.

He also said cool stuff related to money oriented protocol and many more things which can be discussed in depth.

3. A summary on Investigating Commercial Pay-Per-Install and the Distribution of Unwanted Software

In their work, they explore the whole ecosystem of the commercial pay-per-install (PPI) and what is rapid growth of the unwanted software. In CPPI companies bundle their software which can be hidden or useless or unwanted with other software which are more in demand and users want it anyway. Examples: mod application/games, premium version unlocked application and it also could be in the form of the web services and this can be seen in the day to day life use. These PPI based services are based on the regional level pay scale. Based on Google Safe Browsing telemetry, we estimate that PPI networks drive over 60 million download attempts every week—nearly three times that of malware.

They have defined the commercial pay-per-install (PPI) as the practice of software developers bundling several third-party applications in return for a fee.where they have define the PPI PPI Affiliate Structure consists of advertisers, publishers, and PPI affiliate networks which are connected this way Advertiser -> PPI Network <- Publisher -> User.

Commercial PPI networks, like InstallMonetizer and OpenCandy, operate as private companies Where others(Publisher) can register for earning. Registering as a publisher with these networks is straightforward, involving the submission of basic information such as name, website, and estimated daily installs. Underground forums discuss questionable distribution techniques PPI affiliate programs for conversion rates and payouts.

Using the PPI downloader samples they acquire for Amonetize, InstallMonetizer, OpenCandy, and Outbrowse, they develop a pipeline to track the offers that each PPI network distributes as well as the regional price per install.

PPI Downloader Protocol It is like a guidebook which shows how downloaders operate, affecting how software gets spread through PPI networks. In order to track bundled offers, they develop milkers that replay the first stage of each PPI network’s offer protocol and decode the response.

In the longest running campaigns Ad Injectors, Browser Settings Hijackers, System Utilities (Popular like pre installed system manager by third party in new phone), Anti-Virus Products where distribute majorly via PPI and major brands like Opera, Skype and many more also use this PPI. I recently encountered the “Install Opera browser to unlock the game link” situation where the link is broken and they just make fools of people and ask for permissions like use data and send data related to the app to them for future purposes where they are asking for all sorts of the data.

One of the examples of the Pay-Per-Install PPI services is InstallMonetizer which is the platform that allows software developers to monetize their products by bundling them with other software installations.

We also know that in the new phone there are many pre-installed bloatware which are unwanted software which can't be uninstalled which is one of the biggest for now days.

There are many counter measures for individuals and enterprises like IITH to tackle PPI where they can disable the installation of the unwanted software for their local system without permission and for the connected user of IITH they can use the proxy that certainly bans the websites that allow PPI based softwares for the user.

Sources:

https://cs155.stanford.edu/papers/thompson.pdf
https://www.youtube.com/watch?v=qX-ojw1gLmE
https://dl.acm.org/doi/abs/10.5555/3241094.3241151